τ
ARGUS

Version 3.0

User's Manual

Contributors: Anco Hundepool, Aad van de Wetering and Ramya Ramaswamy.
Peter-Paul de Wolf (Modular/Hitas), Sarah Giessing (GHMiter)
Matteo Fischetti, and Juan-José Salazar (Optimisation)
Jordi Castro (Network solutions), Philip Lowthian(manual)

Contents

Preface

This is the user manual for τ-ARGUS version 3.0. τ-ARGUS is a software tool designed to assist a data protector in producing safe tables. This version is the final release of τ-ARGUS in the CASC-project. With respect to the previous release of τ-ARGUS, we have made many steps forward, and τ-ARGUS now has facilities to protect hierarchical and some linked tables.

The purpose of τ-ARGUS is to protect tables against the risk of disclosure, i.e. the accidental or deliberate disclosure of information related to individuals from a statistical table. This is achieved by modifying the table so that it contains less detailed information. τ-ARGUS allows for several modifications of a table: a table can be redesigned, meaning that rows and columns can be combined; sensitive cells can be suppressed and additional cells to protect these can be found in some optimum way (secondary cell suppression).

τ-ARGUS is one of a twin set of disclosure control packages. Within the CASC-project a tool for microdata - called μ-ARGUS - is also being developed, which is the twin brother of τ-ARGUS .[1] This is manifest not only when one looks at the user interfaces of both packages, but also when one looks at the source code: the bodies of the twins are so much combined that they in fact are like Siamese twins!

About the name ARGUS

Somewhat jokingly the name ARGUS can be interpreted as the acronym of 'Anti-Re-identification General Utility System'[2]. As a matter of fact, the name ARGUS was inspired by a myth of the ancient Greeks. In this myth Zeus has a girl friend named Io. Hera, Zeus' wife, did not approve of this relationship and turned Io into a cow. She let the monster Argus guard Io. Argus seemed to be particularly well qualified for this job, because it had a hundred eyes that could watch over Io. If it would fall asleep only two of its eyes were closed. That would leave plenty of eyes to watch Io. Zeus was eager to find a way to get Io back. He hired Hermes who could make Argus fall asleep by the enchanting music on his flute. When Hermes played his flute to Argus this indeed happened: all its eyes closed, one by one. When Hermes had succeeded in making Argus fall asleep, Argus was decapitated. Argus' eyes were planted onto a bird's tail - a type of bird that we now know under the name of peacock. That explains why a peacock has these eye-shaped marks on its tail. This also explains the picture on the cover of this manual. It is a copperplate engraving of Gerard de Lairesse (1641-1711) depicting the process where the eyes of Argus are being removed and placed on the peacock's tail.[3]

Like the mythological Argus, the software is supposed to guard something, in this case data. This is where the similarity between the myth and the package is supposed to end, as we believe that the package is a winner and not a loser as the mythological Argus is.

---

[1] See Anco Hundepool et al., 2004, μ–ARGUS version 4.0 user's manual, Statistics Netherlands, Voorburg, The Netherlands.

[2] This interpretation is due to Peter Kooiman, former head of the methodology department at Statistics Netherlands

[3] The original copy of this engraving is in the collection of 'Het Leidsch Prentenkabinet' in Leiden, The Netherlands.

Contact

Feedback from users will help improve future versions of τ-ARGUS and is therefore greatly appreciated. The authors of this manual can be contacted directly for suggestions that may lead to improved versions of τ-ARGUS in writing or otherwise; e-mail messages can also be sent to argus@cbs.nl.

Acknowledgments

The CASC-project

The CASC project is the initiative in the 5th framework to explore new possibilities of Statistical Disclosure Control and to extend further the existing methods and tools. A key issue in this project is an emphasis more on practical tools, and the research needed to develop them. For this purpose a new consortium has been brought together. It has taken over the results and products emerging from the SDC-project. One of the main tasks of this new consortium was to further develop the ARGUS-software. The main software developments in CASC are μ-ARGUS, the software package for the disclosure control of microdata, while τ-ARGUS handles tabular data.

The CASC-project has involved both research and software development. As far as research is concerned, the project has concentrated on those areas that were expected to result in practical solutions, which can then be built into the software. Therefore the CASC-project has been designed round this software twin ARGUS. This will make the outcome of the research readily available for application in the daily practice of statistical institutes.

CASC-partners

At first sight the CASC-project team had become rather large. However there is a clear structure in the project, defining which partners are working together for which tasks. Sometimes groups working closely together have been split into independent partners only for administrative reasons.

Institute	Short	Country
1. Statistics Netherlands	CBS	NL
2. Istituto Nationale di Statistica	ISTAT	I
3. University of Plymouth	UoP	UK
4. Office for National Statistics	ONS	UK
5. University of Southampton	SOTON	UK
6. The Victoria University of Manchester	UNIMAN	UK
7. Statistisches Bundesamt	StBA	D
8. University La Laguna	ULL	ES
9. Institut d'Estadistica de Catalunya	IDESCAT	ES
10. Institut National de Estadística	INE	ES
11. TU Ilmenau	TUIlm	D
12. Institut d'Investigació en Intelligència Artificial-CSIC	CIS	ES
13. Universitat Rovira i Virgili	URV	ES
14. Universitat Politècnica de Catalunya	UPC	ES

Although Statistics Netherlands is the main contractor, the management of this project is a joint responsibility of the steering committee. This steering committee constitutes of 5 partners, representing the 5 countries involved and also bearing a responsibility for a specific part of the CASC-project:

CASC Steering Committee

Institute	Country	Responsibility
Statistics Netherlands	Netherlands	Overall manager Software development
Istituto Nationale di Statistica	Italy	Testing
Office for National Statistics	UK	
Statistisches Bundesamt	Germany	Tabular data
Universitat Rovira i Virgili	Spain	Microdata

The CASC tabular data team
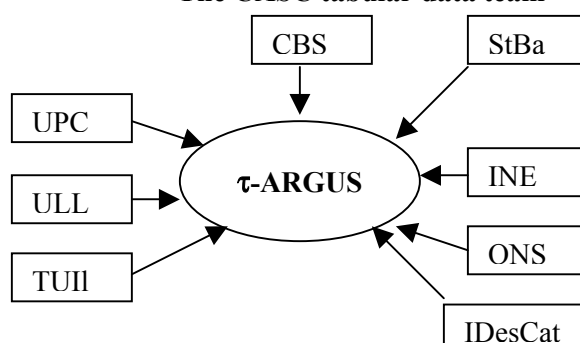
1. Introduction

The growing demands from researchers, policy makers and others for more and more detailed statistical information leads to a conflict. Statistical offices collect large amounts of data for statistical purposes. The respondents are only willing to provide the statistical offices with the required information if they can be certain that these statistical offices will treat their data with the utmost care. This implies that respondents' confidentiality must be guaranteed. This imposes limitations on the amount of detail in the publications. Practice and research have generated insights into how to protect tables, but the problem is not yet definitively solved.

Before we go into more details, the basic ideas on which τ-ARGUS is based, we give a sketch of the general ideas. At first sight one might find it difficult to understand that information presented in tabular form presents a disclosure risk. After all, one might say that the information is presented only in aggregate form.

2. Producing safe tables

Safe tables are produced from unsafe ones by applying certain SDC measures to the tables. These SDC measures - as far as they are implemented in τ-ARGUS - are discussed in the present section. Some key concepts such as sensitive cells, information loss and the like are discussed as well.

2.1 Sensitive cells in magnitude tables

The well-known dominance rule is often used to find the sensitive cells in tables, i.e. the cells that can not be published as they might reveal information on individual records. More particularly, this rule states that a cell of a table is unsafe for publication if a few (n) major contributors to a cell are responsible for a certain percentage (k) of the total of that cell. The idea behind this rule is that in that case at least the major contributors themselves can determine with sufficient precision the contributions of the other contributors to that cell. The choice n=3 and k=70% is not uncommon, but τ-ARGUS will allow the users to specify their own values of n and k.

As an alternative the prior-posterior rule has been proposed. The basic idea is that a contributor to a cell has a better chance to estimate competitors in a cell than an outsider, and also that these kind of intrusions can occur rather often. The precision with which a competitor can estimate is a measure of the sensitivity of a cell. The worst case is that the second largest contributor will be able to estimate the largest contributor. If this precision is more than p%, the cell is considered unsafe. An extension is that also the global knowledge about each cell is taken into account. In that case we assume that each intruder has a basic knowledge of the value of each contributor of q%. Note, that it is actually the ratio p/q that determines which cells are considered safe, or unsafe. In this version of ARGUS, the q-parameter is fixed to 100. Literature refers to this rule as (minimum protection of) p %-rule. If the intention is to state a prior-posterior rule with parameters p_0 and q_0, where $q_0 < 100$, choose the parameter p of the p %-rule as $p = p_0/q_0*100$. See Loeve (2001)[4]

With these rules as a starting point it is easy to identify the sensitive cells, provided that the tabulation package has the facility not only to calculate the cell totals, but also to calculate the number of

---

[4] Loeve, Anneke, 2001, Notes on sensitivity measures and protection levels, Research paper, Statistics Netherlands. Available at http://neon.vb.cbs.nl/casc/related/marges.pdf

contributors and the n individual contributions of the major contributors. Tabulation packages like ABACUS (from Statistics Netherlands) and the package 'SuperCross' developed in Australia by Space-Time Research have that capacity. In fact τ-ARGUS not only stores the sum of the n major contributions for each cell, but the individual major contributions themselves. The reason for this is that this is very handy in case rows and columns etc. in a table are combined. By merging and sorting the sets of individual contributions of the cells to be combined, one can quickly determine the major contributions of the new cell, without going back to the original file. This implies that one can quickly apply the dominance rule to the combined cells. Combining rows and columns (table redesign) is one of the major tools for reducing the number of unsafe cells.

This too is the reason why τ-ARGUS reads microdata files. However due to continuous demands from users we have now also provide the option to read ready-made tables, but with the restriction that the options for table redesign will not then be available.

A problem, however, arises when also the marginals of the table are published. It is no longer enough to just suppress the sensitive cells, as they can be easily recalculated using the marginals. Even if it is not possible to exactly recalculate the suppressed cell, it is possible to calculate an interval that contains the suppressed cell. This is possible if some constraints are known to hold for the cell values in a table. A commonly found constraint is that the cell values are all nonnegative.

If the size of such an interval is rather small, then the suppressed cell can be estimated rather precisely. This is not acceptable either. Therefore it is necessary to suppress additional information to achieve sufficiently large intervals.

Several solutions are available to protect the information of the sensitive cells:
- Combining categories of the spanning variables (table redesign). Larger cells tend to protect the information about the individual contributors better.
- Suppression of additional (secondary) cells to prevent the recalculation of the sensitive (primary) cells.

The calculation of the optimal set (with respect to the loss of information) of secondary cells is a complex OR-problem. τ-ARGUS has been built around this solution, and takes care of the whole process. A typical τ-ARGUS session will be one in which the users will first be presented with the table containing only the primary unsafe cells. The user can then choose how to protect these cells. This can involve the combining of categories, equivalent to the global recoding of μ-ARGUS. The result will be an update of the table with fewer unsafe cells (certainly not more) if the recoding has worked. At a certain stage the user requests the system to solve the remaining unsafe cells by finding secondary cells to protect the primary cells.

At this stage the user can choose between several options to protect the primary sensitive cells. Either they choose the hypercube method or the optimal solution. In this case they also has to select the solver to be used, Xpress or Cplex. After this, the table can be stored for further processing if necessary, and eventual publication.

2.2 Sensitive cells in frequency count tables

In the simplest way of using τ-ARGUS, sensitive cells in frequency count tables are defined as those cells that contain a frequency that is below a certain threshold value. This threshold value is to be provided by the data protector. This way of identifying unsafe cells in a table is the one that is implemented in the current version of τ-ARGUS It should be remarked, however, that this is not always an adequate way to protect a frequency count table.[5] Yet it is applied a lot. Applying a

---

[5] See for instance Leon Willenborg and Ton de Waal, 1996, Statistical disclosure control in practice, Springer-Verlag, New York, Section 6.3.

dominance rule or a p% rule is useless in this context. One should think about possible disclosure risks that a frequency count table poses and possible disclosure scenarios in order to simulate the behaviour of an intruder. Such an analysis would probably come up with different insights than using a simple thresholding rule, *e.g.* like the one sketched in the reference just mentioned. Further research on this topic is being carried out at a.o. Statistics Netherlands.

2.3 Table redesign

If a large number of sensitive cells are present in a table, it might be an indication that the spanning variables are too detailed. In that case one could consider combining certain rows and columns in the table. (This might not always be possible because of publication policy.) Otherwise the number of secondary cell suppressions might just be too enormous. The situation is comparable to the case of microdata containing many unsafe combinations. Rather than eliminating them with local suppressions one can remove them by using global recodings. For tabular data we use the phrase "table redesign" to denote an operation analogous to global recoding in microdata sets. The idea of table redesign is to combine rows, columns etc., by adding the cell contents of corresponding cells from the different rows, columns etc. It is a property of the sensitivity rules that a joint cell is safer than any of the individual cells. So as a result of this operation the number of unsafe cells is reduced. One can try to eliminate all unsafe combinations in this way, but that might lead to an unacceptably high information loss. Instead, one could stop at some point, and eliminate the remaining unsafe combinations by using other techniques such as cell suppression.

2.4 Secondary cell suppression

Once the sensitive cells in a table have been identified, possibly following table redesign it might be a good idea to suppress these values. In case no constraints on the possible values in the cells of a table exist this is easy: one simply removes the cell values concerned and the problem is solved. In practice, however, this situation hardly ever occurs. Instead one has constraints on the values in the cells due to the presence of marginals and lower bounds for the cell values (typically 0). The problem then is to find additional cells that should be suppressed in order to protect the sensitive cells. The additional cells should be chosen in such a way that the interval of possible values for each sensitive cell value is sufficiently large. What is "sufficiently large" can be specified by the data protector in τ-ARGUS by specifying the protection intervals.

In general the secondary cell suppression problem turns out to be a hard problem, provided the aim is to retain as much information in the table as possible, which, of course, is a quite natural requirement. The optimisation problems that will then result are quite difficult to solve and require expert knowledge in the area of combinatorial optimisation.

2.5 Information loss in terms of cell costs

In case of secondary cell suppression it is possible that a data protector might want to differentiate between the candidate cells for secondary suppression. It is possible that they would strongly prefer to preserve the content of certain cells, and are willing to sacrifice the values of other cells instead. A mechanism that can be used to make such a distinction between cells in a table is that of cell costs. In τ-ARGUS it is possible to associate different costs with the cells in a table. The higher the cost the more important the corresponding cell value is considered and the less likely it will be suppressed. We shall interpret this by saying that the cells with the higher associated costs have a higher information

content. The aim of secondary cell suppression can be summarised by saying that a safe table should be produced from an unsafe one, by minimising the information loss, expressed as the sum of the costs associated with the cells that have secondarily been suppressed.

τ-ARGUS offers several ways to compute these costs. The first option is to compute the costs as the sum of the contributions to a cell. Alternatively another variable in the data file can be used as the cost function. Secondly this cost can be the frequency of the contributors to a cell, and finally each cell can have cost = 1, minimising the number of suppressed cells.

2.6 Series of tables

In τ-ARGUS it is possible to specify a series of tables that will be protected one by one, and independently of each other. It is more efficient to choose this option since τ-ARGUS requires only a single run through the microdata in order to produce the tables. But also for the user it is often more attractive to specify a series of tables and let τ-ARGUS protect them in a single session, rather than have several independent sessions.

2.7 The Hypercube/GHMITER method[6]

In order to ensure tractability also of big applications, τ–ARGUS interfaces with the GHM*ITER* hypercube method of R. D. Repsilber of the Landesamt für Datenverarbeitung und Statistik in Nordrhein-Westfalen/Germany, offering a quick heuristic solution. The method has been described in depth in [1], [2] and [3], for a briefer description see [4].

2.7.1 The method

The approach builds on the fact that a suppressed cell in a simple n-dimensional table without substructure cannot be disclosed exactly if that cell is contained in a pattern of suppressed, nonzero cells, forming the corner points of a hypercube.

The algorithm subdivides n-dimensional tables with hierarchical structure into a set of n-dimensional sub-tables without substructure. These sub-tables are then protected successively in an iterative procedure that starts from the highest level. Successively, for each primary suppression in the current sub-table, all possible hypercubes with this cell as one of the corner points are constructed.

For each hypercube, a lower bound is calculated for the width of the suppression interval for the primary suppression, that would result from the suppression of all corner points of the particular hypercube. To compute that bound, it is not necessary to implement the time consuming solution to the Linear Programming problem. If it turns out that the bound is sufficiently large, the hypercube becomes a feasible solution. For any of the feasible hypercubes, the loss of information associated with the suppression of its corner points is calculated. The particular hypercube that leads to minimum information loss is selected, and all its corner points are suppressed.

After all sub-tables have been protected once, the procedure is repeated in an iterative fashion. Within this procedure, when cells belonging to more than one sub-table are chosen as secondary suppressions in one of these sub-tables, in further processing they will be treated like sensitive cells in the other sub-tables they belong to. The same iterative approach is used for sets of linked tables.

---

[6] The section on GHMiter has been contributed by Sarah GIESSING, *Federal Statistical Office of Germany 65180 Wiesbaden E-mail:* sarah.giessing@destatis.de

---

It should be mentioned here that the 'hypercube criterion' is a sufficient but not a necessary criterion for a 'safe' suppression pattern. Thus, for particular subtables the 'best' suppression pattern may not be a set of hypercubes – in which case, of course, the hypercube method will miss the best solution and lead to some overprotection. Other simplifications of the heuristic approach that add to this tendency for over-suppression are the following: when assessing the feasibility of a hypercube to protect a specific target suppressions against interval disclosure, the method

- is not able to consider protection possibly already provided by other cell suppressions (suppressed cells that are not corner points of this hypercube) within the same sub-table,
- does not consider the sensitivity of multi-contributor primary suppressions properly, that is, it does not consider the protection already provided in advance of cell suppression through aggregation of these contributions,
- attempts to provide the same *relative* ambiguity to (eventually large) secondary suppressions that have been selected to protect cells in a linked sub-table, as if they were single-respondent primary suppressions, while actually it would be enough to provide the same *absolute* ambiguity as required by the corresponding primary suppressions.

2.7.2 The ARGUS implementation of GHMITER

- In the implementation offered by ARGUS, GHMITER makes sure that a single respondent cell will never appear to be corner point of one hypercube only, but of two hypercubes at least. Otherwise it could happen that a single respondent, who often can be reasonably assumed to know that he is the only respondent, could use his knowledge on the amount of his own contribution to recalculate the value of any other suppressed corner point of this hypercube.
- For tables presenting magnitude data, τ-ARGUS will ensure that GHMITER selects secondary suppressions that protect the sensitive cells properly, at least to the extent possible. It is assumed that users of the table can estimate any cell value to within some percentage of its actual value in advance of the publication, the so called *a priori* bound. By default τ-ARGUS assumes this percentage to be 100, but the user is offered to change it in the screen below:

Considering the given *a priori* bounds, τ-ARGUS will compute a suitable *sliding protection ratio* (for explanation see [5], τ-ARGUS will display the value of this ratio in the report file) to be used by GHMITER, to make it select secondary suppressions that are sufficiently large. This approach ensures that a user of the resulting protected table when using, apart from the assumed *a priori* information, only information provided by the data of the protected table, would normally not be able to derive any bounds for the contribution of any respondent to a particular sensitive cell close enough to disclose this contribution according to the primary sensitivity rule in use.

- Note, if in the screen above the option "*Protection against inferential disclosure required*" is inactivated, GHMITER will not check whether secondary suppressions are sufficiently large.
- As mentioned above, GHMITER is unable to 'add' the protection given by multiple hypercubes. In certain situations, considering the given *a priori* bounds, it is not possible to provide sufficient protection to a particular sensitive cell (or secondary suppression) by suppression of one single hypercube. In such a case, GHMITER is unable to confirm that this cell has been protected properly, according to the specified *sliding protection ratio*. It will then reduce the *sliding protection ratio* automatically, and individually, step by step for those cells, the protection of which the program cannot confirm otherwise. In steps 1 to 9 we divide the original ratio by k,

values of k from 2 to 10, and if this still does not help, in step 10 we divide by an extremely large value, and finally, if even that does not solve the problem, step 11 will set the ratio to zero). The τ-ARGUS report file will display the number of cases where the sliding protection range was reduced, by finally confirmed sliding protection ranges.

Note that that the number of cases with range reduction reported by this statistic in the report file is very likely to exceed the actual number of cells concerned, because cells belonging to multiple (sub-) tables are counted multiple times. In our experience this concerns particularly the cases, where the protection level was reduced to an 'infinitely' small (positive) value (in step 10, see above). Step 10 is usually required to confirm protection of large, high level secondary suppressions, which are likely to appear in multiple tables, especially in processing of linked tables. By the way, terms "reduction of the *sliding protection ratio*" and "reduction of the *protection level*" are used synonymously in the report file.

- Note that step 11 will make cells eligible for secondary suppression that τ-ARGUS considers as 'protected' (so called '*frozen*' cells, for discussion of this option see for instance [5]).

As this is inconsistent with the current view on protected cells in τ-ARGUS this will lead to the following error message:
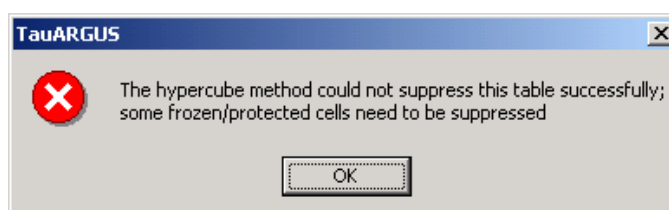
Codes and cell values of those suppressed *frozen* cells are then displayed by ARGUS:
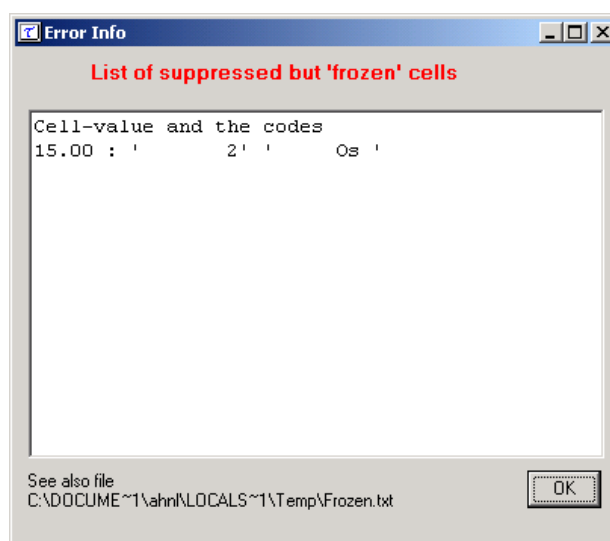
When the status of these cells is changed into 'unprotected' before re-running the hypercube method, the solution will be a feasible solution for τ-ARGUS.

2.7.3 References on GHMiter

[1]   Repsilber, R. D. (1994), 'Preservation of Confidentiality in Aggregated data', paper presented at the Second International Seminar on Statistical Confidentiality, Luxembourg, 1994

[2]   Repsilber, D. (1999), 'Das Quaderverfahren' - in *Forum der Bundesstatistik, Band 31/1999: Methoden zur Sicherung der Statistischen Geheimhaltung*, (in German)

[3]   Repsilber, D. (2002), 'Sicherung persönlicher Angaben in Tabellendaten' - in *Statistische Analysen und Studien Nordrhein-Westfalen, Landesamt für Datenverarbeitung und Statistik NRW,* Ausgabe 1/2002 (in German)

[4]   Giessing, S. and Repsilber, D. (2002), 'Tools and Strategies to Protect Multiple Tables with the GHQUAR Cell Suppression Engine', in '*Inference Control in Statistical Databases*' Domingo-Ferrer (Editor), Springer Lecture Notes in Computer Science Vol. 2316.

[5]   Giessing, S. (2003), 'Co-ordination of Cell Suppressions: strategies for use of GHM*ITER*', Proceedings of the Joint ECE/Eurostat work session on statistical data confidentiality (Luxembourg, 7-9 April 2003)

2.8 Optimisation models for secondary cell suppression[7]

τ-ARGUS applies different approaches to find optimal and near-optimal solutions. One of these approaches is based on a Mathematical Programming technique which consists of solving Integer Linear Programming programs modelling the combinatorial problems under different methodologies (Cell Suppression and Controlled Rounding). The main characteristic of these models is that they share the same structure, thus based only on a 0-1 variable for each cell. In the Cell Suppression methodology, the variable is 1 if and only if the cell value must be suppressed. In the Controlled Rounding methodology, the variable is 1 if and only if the cell value must be rounded up. No other variables are necessary, so the number of variables in the model is exactly the number of cells in the table to be protected. In addition, the model also imposes the protection level requirements (upper, lower and sliding) in the same way for the different methodologies (Cell Suppression and Controlled Rounding). These requirements ask for a guarantee that an attacker will not get too narrow an interval of potential values for a sensitive cell, which he/she will compute by solving two linear programming programs (called attacker problems). Even if a first model containing this two-attacker problem would lead to a bi-level programming model, complex to be solved in practice, a Benders' decomposition approach allows us to convert the attacker problems into a set of linear inequalities. This conversion provides a second model for each methodology that can be efficiently solved by a modern cutting-plane approach. Since the variables are 0-1, a branching phase can be necessary, and the whole approach is named "branch-and-cut algorithm".

Branch-and-cut algorithms are modern techniques in Operations Research that provide excellent results when solving larger and complicated combinatorial problems arising in many applied fields (like routing, scheduling, planning, telecomunications, etc.). Shortly, the idea is to solve a compact 0-1 model containing a large number of linear inequalities (as the ones above mentioned for the Cell Suppression and for the Controlled Rounding) through an iterative procedure that does not consider all the inequalities at the same time, but generates the important ones when needed. This dynamic procedure of dealing with large models allows the program to replace the resolution of a huge large model by a short sequence of small models, which is termed a "decomposition approach". The on-line generation of the linear inequalities (rows) was also extended in this work to the variables (columns), thus the algorithm can also works on tables with a large number of cells, and the overall algorithm is named "branch-and-cut-and-price" in the Operations Research literature.

---

[7] The optimisation models have been built by a team of researchers headed by Juan-José Salazar-Gonzalez of the University La Laguna, Tenerife, Spain. Other members of the team were: G. Andreatta, M. Fischetti, R. Betancort Villalva, M.D. Montesdeoca Sanchez and M. Schoch

To obtain good performance, the implementation has also considered many other ingredients, standard in branch-and-cut-and-price approaches. For example, it is fundamentally the implementation of a preprocessing approach where redundant equations defining the table are eliminated, where variables associated to non-relevant cells are removed, and where dominated protection levels are detected. The preprocessing is fundamental to make the problem as small as possible before starting the optimization phase. Another fundamental ingredient is the heuristic routine, which allows the algorithm to start with an upper bound of the optimal loss of information. This heuristic routine ensures the production of a protected pattern if the algorithm is interrupted by the user before the end. In other words, thanks to the heuristic routine, the implemented algorithm provide a near-optimal solution if the execution is cancelled before having a proof of optimality. During the implicit enumeration approach (i.e., the branch-and-cut-and-price) the heuristic routine is called several times, thus providing different protected patterns, and the best one will be the optimal solution if its loss of information is equal to the lower bound. This lower bound is computed by solving a relaxed model, which consists of removing the integrability condition on the integer model. Since the relaxed model is a linear program, a linear programming solver must be called.

We have not implemented out own linear programming solver, but used a commercial solver which is already tested by other programmers for many years. A robust linear programming solver is a guarantee that no numerical trouble will appear during the computation.

That is the reason to required either CPLEX (from ILOG) or XPRESS (from DashOptimization). Because the model to be solved can be applied to all type of table structures (2-dim, 3-dim, 4-dim, 5-dim, etc), including hierarchical and linked tables, we cannot use special simplex algorithm implementations, like the min-cost flow computation which would required to work with tables that can be modelled as a network (e.g., 2-dimensional tables or collections of 2-dim tables linked by one link). On this special table, ad-hoc approaches (solving network flows or short path problems) could be implemented to avoid using general linear programming solvers.

In any case, future works will try to replace the commercial solvers by freely available linear-programming solvers.

2.9 The Modular approach

The modular (HiTaS) solution is a heuristic approach to cell suppression in hierarchical tables. Hierarchical tables are specially linked tables: at least one of the spanning variables exhibits a hierarchical structure, *i.e.* contains (many) sub-totals.

In Fischetti and Salazar (1998) a theoretical framework is presented that should be able to deal with hierarchical and generally linked tables. In what follows, this will be called the mixed integer approach. In this framework, additional constraints to a linear programming problem are generated. The number of added constraints however, grows rapidly when dealing with hierarchical tables, since many dependencies exist between all possible (sub-)tables containing many (sub-)totals. The implemented heuristic approach (HiTaS) deals with a large set of (sub-)tables in a particular order. A non hierarchical table can be considered to be a hierarchical table with just one level. In that case, the approach reduces to the original mixed integer approach and hence provides the optimal solution. In case of a hierarchical table, the approach will provide a sub-optimal solution that minimises the information loss per sub-table, but not necessarily the global information loss of the complete set of hierarchically linked tables.

In the following section, a short description of the approach is given. For a more detailed description of the method, including some examples, see *e.g.*, De Wolf (2002).

HiTaS deals with cell suppression in hierarchical tables using a top-down approach. The first step is to determine the primary unsafe cells in the base-table consisting of all the cells that appear when crossing the hierarchical spanning variables. This way all cells, whether representing a (sub-)total or

not, are checked for primary suppression. Knowing all primary unsafe cells, the secondary cell suppressions have to be found in such a way that each (sub-)table of the base-table is protected and that the different tables cannot be combined to undo the protection of any of the other (sub-)tables. The basic idea behind the top-down approach is to start with the highest levels of the variables and calculate the secondary suppressions for the resulting table. The suppressions in the interior of the protected table is then transported to the corresponding marginal cells of the tables that appear when crossing lower levels of the two variables. All marginal cells, both suppressed and not suppressed, are then 'fixed' in the calculation of the secondary suppressions of that lower level table, i.e., they are not allowed to be (secondarily) suppressed. This procedure is then repeated until the tables that are constructed by crossing the lowest levels of the spanning variables are dealt with.

A suppression pattern at a higher level only introduces restrictions on the marginal cells of lower level tables. Calculating secondary suppressions in the interior while keeping the marginal cells fixed, is then independent between the tables on that lower level, i.e., all these (sub)-tables can be dealt with independently of each other. Moreover, added primary suppressions in the interior of a lower level table are dealt with at that same level: secondary suppressions can only occur in the same interior, since the marginal cells are kept fixed.

However, when several empty cells are apparent in a low level table, it might be the case that no solution can be found if one is restricted to suppress interior cells only. Unfortunately, backtracking is then needed.

Obviously, all possible (sub)tables should be dealt with in a particular order, such that the marginal cells of the table under consideration have been protected as the interior of a previously considered table. To that end, certain groups of tables are formed in a specific way (see De Wolf (2002)). All tables within such a group are dealt separately, using the mixed integer approach.

The number of tables within a group is determined by the number of parent-categories the variables have one level up in the hierarchy. A parent-category is defined as a category that has one or more sub-categories. Note that the total number of (sub)-tables that have to be considered thus grows rapidly.

Singletons

Singleton cells should be treated with extra care. The single respondent in this cell could easily undo the protection if no extra measures were taken. The most dangerous situation is that there are only two singletons in a row, or one singleton and one other primary unsafe cell. These singletons could easily disclose the other cell.

In the current implementation, we have made sure that at least two singletons in one row or column cannot disclose each other's information. For this, we increase the protection margins of these singletons such that the margin of the largest is greater than the cell-value of the smallest.

References on the modular method

Fischetti, M. and J.J. Salazar-González (1998). *Models and Algorithms for Optimizing Cell Suppression in Tabular Data with Linear Constraints.* Technical Paper, University of La Laguna, Tenerife.

P.P. de Wolf (2002). *HiTaS: a heuristic approach to cell suppression in hierarchical tables.* Proceedings of the AMRADS meeting in Luxembourg (2002).

Additional reading on the optimisation models can be found at the CASC-website

(http://neon.vb.cbs.nl/casc/Related/99wol-heu-r.pdf)

2.10 *Network solution for large 2 dimensional tables with one hierarchy*

τ-ARGUS also contains a solution for the secondary cell suppression based on network flows. This contribution is by Jordi Casto of the Universitat Politècnica de Catalunya in Barcelona. The network flows solution for cell suppression implements a fast heuristic for the protection of statistical data in two-dimensional tables with one hierarchical dimension (1H2D tables). This new heuristic sensibly combines and improves ideas of previous approaches for the secondary cell suppression problem in two-dimensional general [2] and positive [7, 9] tables. Details about the heuristic can be found in [4, 5]. Unfortunately this approach is only possible for two-dimensional tables with only one hierarchy, due to the limitations of the network flows.

The heuristic is based on the solution of a sequence of shortest-path subproblems that guarantee a feasible pattern of suppressions (i.e., one that satisfies the protection levels of sensitive cells). Hopefully, this feasible pattern will be close to the optimal one.

The current package is linked with three solvers: CPLEX7.5/8.0 [8], PPRN [6], and an efficient implementation of the bidirectional Dijkstra's algorithm for shortest-paths (that will be denoted as "Dijkstra") [1]. Later releases of CPLEX will also work if the interface routines are the same than for version 8.0. The heuristic can use any of the three solvers for the solution of the shortest path subproblems, although Dijkstra is recommended (and the default one) for efficiency reasons. CPLEX is needed if a lower bound of the optimal solution want to be computed. The auditing phase can be performed with either CPLEX or PPRN.

PPRN and Dijkstra were implemented at the Dept. of Statistics and Operations Research of the Universitat Politècnica de Catalunya, and are included in NF CSP. PPRN was originally developed during 1992–1995, but it had to be significantly improved within the CASC project to work with NF CSP. Dijkstra was completely developed in the scope of CASC. The third solver, CPLEX, is a commercial tool, and requires purchasing a license. However, PPRN is a fairly good replacement— although not so robust— for the network flows routines of CPLEX. Therefore, in principle, there is no need for an external commercial solver, unless lower bounds want to be computed.

Even though two of the three solvers are included in the distribution of NF CSP, this document only describes the features of the heuristic, and from the user's point of view. A detailed description of PPRN and Dijkstra's solvers can be found in [3, 6] and [1], respectively.

The current implementation in τ-ARGUS however only uses the Dijkstra and the PPRN solvers. We have restricted ourselves from commercial solvers here as the network flows give already a very fast solution.

References on the network solution
[1] Ahuja, R.K, Magnanti, T.L., Orlin, J.B., Network Flows, Prentice Hall (1993).
[2] Castro, J., PPRN 1.0, User's Guide, Technical report DR 94/06 Dept. of Statistics and Op-erations Research, Universitat Politècnica de Catalunya, Barcelona, Spain, 1994.
[3] Castro, J., Network flows heuristics for complementary cell suppression: an empirical evaluation and extensions, in LNCS 2316, Inference Control in Statistical Databases, J. Domingo-Ferrer (Ed), (2002) 59–73.
[4] Castro, J., Nabona, N. An implementation of linear and nonlinear multicommodity network flows. European Journal of Operational Research 92, (1996) 37–53.
[5] Cox, L.H., Network models for complementary cell suppression. J. Am. Stat. Assoc. 90, (1995) 1453–1462.
[6] ILOG CPLEX, ILOG CPLEX 7.5 Reference Manual Library, ILOG, (2000).
[7] Kelly, J.P., Golden, B.L, Assad, A.A., Cell Suppression: disclosure protection for sensitive tabular data, Networks 22, (1992) 28–55.
[8] Castro, J. User's and programmer's manual of the network flows heuristics package for cell suppression in 2D tables Technical Report DR 2003-07, Dept. of Statistics and Operations Research, Universitat Politècnica de Catalunya, Barcelona, Spain,2003;

See http://neon.vb.cbs.nl/casc/deliv/ 41D6_NF1H2D-Tau-Argus.pdf

2.11 Functional design of τ-ARGUS
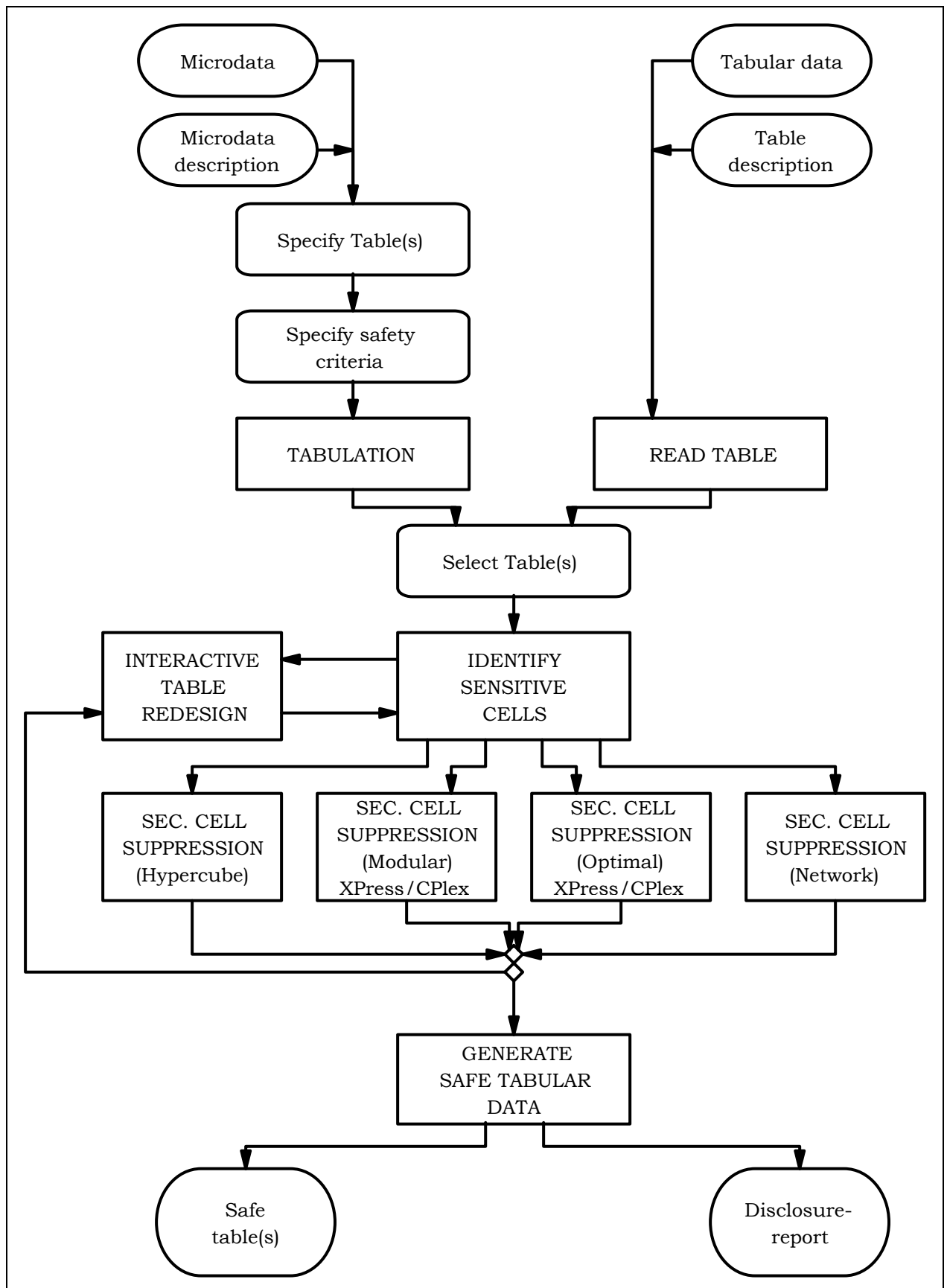
3. A tour of τ-ARGUS

In this chapter, we explain and display the key features of τ-Argus. τ-Argus is a menu driven program, and here we describe a number of menus which the user will follow in order to prepare a table for output in a 'safe' form. The aim of the tour is to guide the user through the basic features of the program without describing every feature in detail. The only pre-requisite knowledge is basic experience of the Windows environment. In Chapter 4 (Reference) a more systematic description of the different parts of τ-ARGUS will be given. Chapter 3 can be read as a standalone chapter as there is enough detail to enable the user to run the program. However, not every option is covered and the user is pointed in the direction of the Reference chapter in a number of instances. In addition, back references to the theory explained in Chapter 2 are also indicated. In this tour we will use the data in the file tau_testW.asc, which comes with the installation of τ-ARGUS.

The key windows for preparation of the data and the processes of disclosure control (depicted graphically in the figure in section 2.11) are explored in this tour, which are given below.

Preparation

Open Microdata. This involves declaring both the microdata and the associated metadata.

Specify Metafile. This shows how the metafile can be edited after being read in but before any tables have been specified. This includes options such as declaring variables to be explanatory or response, and setting up the hierarchical structure of the data.

Specify Tables. Declare the tables for which protection is required, along with the safety rule and minimum frequency rule on which the primary suppressions will be based.

Process of Disclosure Control

View Table. View the table after the safety rules for primary suppressions have been applied. This is a key window in which the possible options to make the table safe are discussed such as recoding and the application of the required method of secondary suppression.

Save Table. The user can save the 'safe' table in a number of formats as will be seen in section 3.2.2.

3.1 Preparation

To start disclosure control with τ-ARGUS there are two possible options:

1. Open a microdata file from which a table can be constructed,

2. Open an already-constructed table.

In this tour we only deal with how to open a fixed format microdata file (sections 3.1.1 to 3.1.3). If an already constructed table is to be used, then go to the Reference chapter (section 4.2.2). Some methods

for secondary suppression (the modular and the optimal) require an external linear programming solver. The choice of this solver can be decided before opening a dataset. The choices are either Xpress or Cplex, the different implementations of the search-algorithms described in the Theory chapter (section 2.8). This information can be supplied by clicking on (Help|Options) to give the following window.
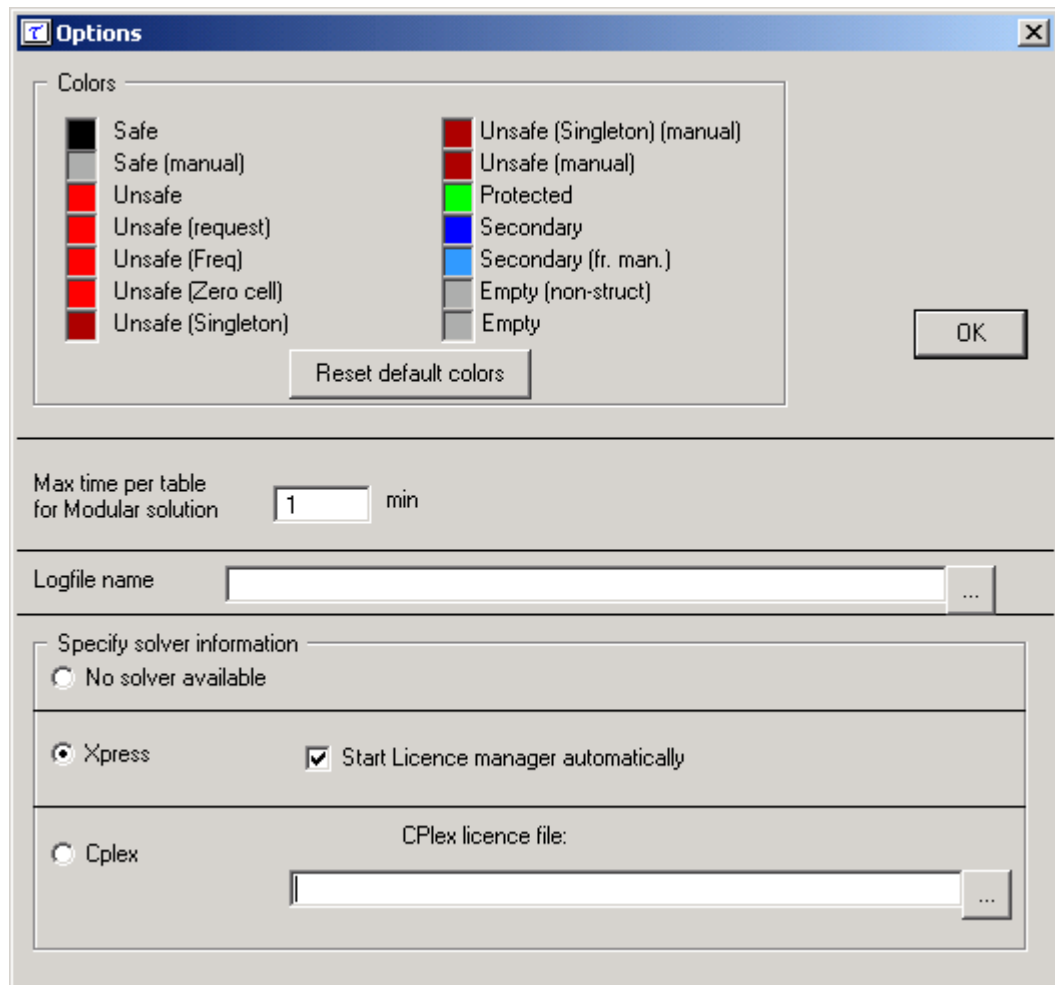
Once this window has been opened details of the solver can be entered. Other alterations to the default such as the colour of values in particular cells can also be made.

3.1.1 Open a microdata file

Both a microdata file and the metadata file describing this microdata file are required. The microdata file must be either a fixed format ASCII file or a free format file with a specified separator. By clicking (File|Open Microdata) you can specify both the name of the microdata file and the name of the file containing the metadata.
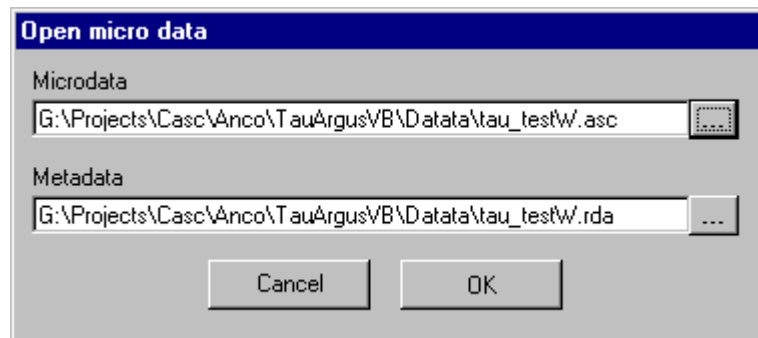
For most uses of τ-Argus, the microdata and metadata file are stored in separate files. The simplest way to use the program is to use the extension .ASC for the datafile and .RDA for the metadata file. If the name of the metadata file is the same as the datafile, except for the extension, and it already exists in the same directory, τ-ARGUS will fill in the name of this file automatically in the space under the metadata heading. If no metadata file is specified, the program has the facility to let you specify the metadata interactively via the menu option (Specify|Metafile). This is also the place to make changes to the metadata file. In subsection 3.1.2 we will give a description of the metadata file for τ-ARGUS.

3.1.2 Specify metafile

When you enter or change the metadata file interactively using τ-ARGUS the option (Specify|Metafile) will bring you to the following screen:
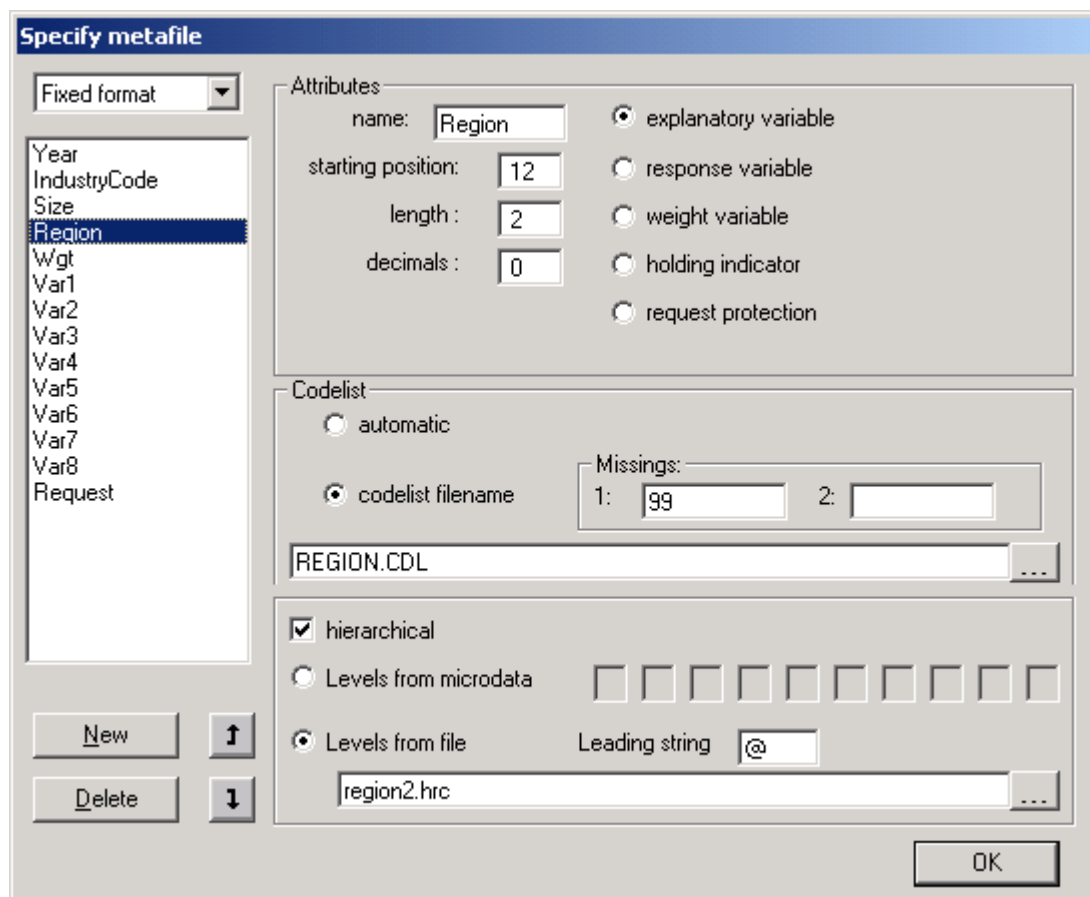
The key elements of this window are the definitions for each variable. Most variables will be defined as one of the following.

- Explanatory Variable: a variable to be used as a categorical (spanning) variable when defining a table.
- Response Variable: a variable to be used as a cell item in a table.
- Weight variable: a variable containing the sampling weighting scheme.

More details on these variables along with the others can be found in the Reference chapter (subsection 4.3.1).

Other important features of this window are as follows.

- Codelist: τ-ARGUS will automatically build the codelists for the explanatory variables, or you can specify a codelist file (a list-of-codes of the explanatory variables) as follows.
  - Automatic: The codelist is created from the categories in the variable.
  - Codelist file: The codes can be read in from an external file. Each category can contain a label. The codelist is only used for enhancing the presentation but always τ-ARGUS will build a codelist from the datafile itself.
- Missing values: this gives information on the missing values which are attached to a codelist. Two distinct missing value indicators can be set (the reason for this is for the purposes of indicating different reasons for missing values: for example perhaps non-responses of different forms: maybe one code for the response '*don't know*', and another for '*refusal*'). Missing values however are not required.
- Hierarchical codes: The hierarchy can be derived from
  1. The digits of the individual codes in the data file or
  2. A specified file containing the hierarchical structure.

Examples are shown in the metafile information below.

The Metafile
The metafile describes the variables in the microdata file, both the record layout and some additional information necessary to perform the SDC-process. Each variable is specified on one main line, followed by one or more option lines. An example is shown here. The leading spaces shown only serve only to make the file more readable; they have no other meaning.

```
Year 1  2 99
   <RECODEABLE>
IndustryCode 4  5 99999
   <RECODEABLE>
   <HIERARCHICAL>
   <HIERLEVELS>  3 1 1 0 0
Size 9  2 99
   <RECODEABLE>
Region 12  2 99
   <RECODEABLE>
   <CODELIST> Region.cdl
   <HIERCODELIST>  Region2.hrc
   <HIERLEADSTRING> @
   <HIERARCHICAL>
Wgt 14  4 9999
   <NUMERIC>
   <DECIMALS>  1
   <WEIGHT>
Var1 19  9 999999999
```

```
   <NUMERIC>
Var2 28  10 9999999999
   <NUMERIC>
   <DECIMALS>  2
……
……
```

Details of the variables

'Year' : For this variable each record begins on position 1, is 2 characters long and missing values are represented by 99. It is also recodeable implicitly stating that it is an explanatory or spanning variable used to create the tables.

'IndustryCode': For this variable each record begins on position 4 and is 5 characters long. Missing values are represented by 99999. As well as being recodeable this variable is hierarchical and the hierarchy structure is specified. The first 3 characters are in the top hierarchy level, the 4th character in the second level and the 5th character in the lowest level. The two zeros at the end of this definition are redundant in this example.

'Size': For this variable each record begins on position 9 and is 2 characters long, and missing values are represented by 99. It is also recodeable.

'Region': For this variable each record begins on position 12 and is 2 characters long. Missing values are represented by 99. An example of a codelist file can be found in *region.cdl* and of a hierarchical codelist file in *region2.hrc*. Contents of these files are shown here.

The file region.cdl:
```
 1,Groningen
 2,Friesland
 3,Drenthe
 4,Overijssel
 5,Flevoland
 6,Gelderland
 7,Utrecht
 8,Noord-Holland
 9,Zuid-Holland
10,Zeeland
11,Noord-Brabant
12,Limburg
Nr,North
Os,East
Ws,West
Zd,South
```

The file region.hrc:
```
Nr
@ 1
@ 2
@ 3
Os
@ 4
@ 5
@ 6
@ 7
```

```
Ws
@ 8
@ 9
@10
Zd
@11
@12
```

Additional details of these coding files can be found in the Reference chapter (section 4.2.1).

3.1.3 Specify tables

When the metadata file is ready, the tables to be protected can be specified. This is achieved via Specify|Specify Tables. A window to specify the tables is presented. In the example here we have a 2 dimensional table (2 explanatory variables) and a single response variable. A safety rule has been defined.

The key elements of this window are as follows.

Explanatory variables
On the left is the listbox with the explanatory variables.

Click on '>' the selected variable to the next box in which the selected explanatory variables can be seen. From the box on the left hand side, containing explanatory variables, the variables that will be used in the row or the column of the table, in a 2-way table can be selected. Up to four explanatory variables can be selected to create a table.

Cell items

The 'cell items' box contains the variables, which were declared as 'response variables' in the metafile. By using the '>' button they can be moved to the 'response variable' box to be used in the defined table.

Response variable
Any variable in the cell items box can be chosen as the response variable. More than one response variable can be chosen.

Shadow variable
The shadow variable is the variable which is used to apply the safety rule. By default this is the response variable. More detail on the Shadow variable can be found in section 4.3.3 in the Reference chapter.

Cost variable
This variable describes the cost of each cell. These are the costs that are minimised when the secondary suppressed cells are calculated (See section 2.5 in the Theory chapter for the further details). By default this is the response variable but other choices are possible. If the response or any other explicitly specified variable is used for this purpose, the circle next to 'variable' should be filled. Then, any variable name can be transferred from the cell items to the cost variable window. It is also possible to use the frequency of the cells as a cost-function. This will suppress cells with respect to number of contributors to each cell. A third option is that the number of cells to be suppressed is minimised, irrespective of the size of their contributions (unity option – cost variable is set to 1 for each cell). More details will be given in the Reference Chapter along with an example (section 4.3.3).

Weight
If the data file has a sample weight, specified in the metadata file, the table can be computed taking this weight into account. In this case, the 'apply weights' box should be ticked. More details will be given in the Reference Chapter along with an example (section 4.3.3).

The safety rule
The concept of safety rules is explained in section 2.1 in the chapter on Theory. In this window the left side of the window allows the type of rule to be selected, this is usually either the dominance rule or p% rule, along with the necessary parameter values. Several rules together can be set for any particular table.

Additionally, the minimum number of contributors (threshold rule) can be chosen. In the window this is referred to as the 'Minimum Frequency'

Now, brief summaries are provided to define the Dominance and p% rules.

Dominance Rule
This is sometimes referred to as the (n,k) rule where n is the number of contributors to a cell contributing more than k% of the total value of the cell (if the cell is to defined as unsafe for publication).

p% rule
The p% rule says that if x_1 can be determined to an accuracy of better than p% of the true value then it is disclosive where x_1 is the largest contributor to a cell.
This rule can be written as:

$$\sum_{i=3}^{c} x_i \geq \frac{p}{100} x_1$$ for the cell to be non-disclosive where c is the total number of contributors to the cell

and the intruder is a respondent in the cell.

It is important to know that when entering this rule in τ-ARGUS the value of *n* refers to the number of intruders in coalition (who wish to group together to estimate the largest contributor).

A typical example would be that the sum of all reporting units excluding the largest two must be at least 10% of the value of the largest. Therefore, in τ-Argus set p=10 and n =1 as there is just one intruder in the coalition, respondent x_2.

The choice of safety rule is specified by the user and the chosen parameters can then be entered. From these parameters symmetric safety ranges are computed automatically prior to the secondary suppressions.

For the minimum frequency rule, a safety range is calculated from the user given range. This is usually a small positive value and is required to enable secondary suppression to be carried out.
A manual safety range is also required for cells that can be made unsafe by intervention of the user.
Other options such as the 'Request Rule' or the 'Holding Rule' will be looked at in more detail in the Reference chapter (section 4.3.3).

When everything has been filled in, click '∨' to transport all the specified parameters describing tey table to the 'listwindow' on the bottom. As many tables as you want may be specified, only limited by the memory of the computer. If a table is to be modified press the '^' button.

Creating the Table

Pressing the 'Compute tables' button will invoke τ-ARGUS to actually compute the tables requested and the process to start disclosure control may be invoked. τ-ARGUS will come back with the main window showing the number of unsafe cells per variable, per dimension, as explained in the next section 3.2.

3.2 The Process of disclosure control

When the table(s) have been calculated, the main-window of τ-ARGUS will be displayed again with an overview of all the unsafe cells per variable (over all the tables). An example is shown here. This window (underneath the main menu for τ-ARGUS), shows the number of unsafe combinations per variable. For example there are no single unsafe cells in dimension one for either variable. (i.e. the

one-way marginal total for different values of 'Size' and 'Region' are all not disclosive).

The right hand window gives the equivalent information for each level of the variable indicated on the left. For example, there are 12 unsafe cells in the two way 'Size' x 'Region' table.

Size

There are however 12 unsafe cells in the 2 way table *Size* by *Region* as can be seen by the right hand window which gives the equivalent information for each level of the variable indicated on the left. There are 5 unsafe cells where Size = 2, 6 unsafe cells where Size = 4 and a single unsafe cell where Size = 9.

Region

Two of these are 'North' subtotal cells. Within the 'North' region, 2 cells in Groningen are disclosive. Two 'East' subtotal cells are also unsafe. Within the 'East' region 2 cells in 'Overijssel' are unsafe along with 2 cells in 'Gelderland'.   Finally there is 1 unsafe cell for the 'South' subtotal and within this region there is 1 unsafe cell for 'Noord-Brabant'

3.2.1 View table

The 'Modify|View table' option shows the calculated table plus some additional information. This is regarded as the key window in τ-Argus.

The selected table is displayed in a spreadsheet view. Safe cells are shown in black, whilst cells failing the safety rule and/or minimum frequency rule are displayed in red. These are the default colours.

The user now has to decide whether to carry out secondary suppressions immediately or to perform some recoding first. There are other options such as changing the status of individual cells manually, this will be discussed further in the Reference chapter (see section 4.4.2).

Table: Size x Region | Var2

	tot	2	4	5	6	7	8	9	99
tot	16,847,647	20	25	2,711,808	2,320,534	2,505,043	2,799,074	6,510,758	385
Nr	4,373,664	5	5	719,049	659,680	688,962	756,529	1,549,049	385
1	1,986,129	5	5	398,062	348,039	354,711	418,778	466,529	-
2	1,809,246	0	-	223,990	221,332	241,913	258,233	863,393	385
3	578,289	-	-	96,997	90,309	92,338	79,518	219,127	-
Os	3,703,896	15	5	642,238	515,003	534,147	620,392	1,392,096	-
4	124,336	5	-	36,311	32,132	25,770	18,150	11,968	-
5	526,279	-	-	93,589	94,957	110,930	81,799	145,004	-
6	2,234,995	10	5	345,803	251,358	251,188	303,377	1,083,254	-
7	818,286	-	-	166,535	136,556	146,259	217,066	151,870	-
Ws	4,576,116	-	-	648,972	543,570	663,897	775,132	1,944,545	-
8	485,326	-	-	63,767	75,442	87,305	59,953	198,859	-
9	3,664,560	-	-	537,911	430,851	515,020	643,762	1,537,016	-
10	426,230	-	-	47,294	37,277	61,572	71,417	208,670	-
Zd	4,193,971	-	15	701,549	602,281	618,037	647,021	1,625,068	-
11	2,752,743	-	15	488,613	392,395	363,490	402,925	1,105,305	-
12	1,441,228	-	-	212,936	209,886	254,547	244,096	519,763	-
99	-	-	-	-	-	-	-	-	-

Cell Information

- Value: 16,847,647
- Status: Safe
- Cost: 16,847,647
- Shadow: 16,847,647
- # contributions: 42723
- Top n of shadow: 175,677 / 141,482 / 135,469
- [ ] Holding level
- Request: 0

Change status

Set to Safe | Set to Unsafe | A priori info | Set to Protected

Recode

Suppress

(•) HyperCube | Singleton
( ) Modular | Undo Singleton
( ) Network
( ) Optimal | Suppress | Undo Suppress | Audit

[x] 3 dig. separator | Select Table | Change View | Write table
[ ] Output View | Table Summary | Close

Cell information

Cells can be selected in the table by moving the cursor arrow. In each case, information about the selected cell is shown on the right.

The status of the cell can be one of the following. Some of the terms will be explained later in this section but others are expanded upon in the Reference section 4.4.2.

- Safe: Does not violate the safety rule
- Safe (from manual): manually made safe during this session
- Unsafe: According to the safety rule
- Unsafe (request): Unsafe according to the Request rule.
- Unsafe (frequency): Unsafe according to the minimum frequency rule.
- Unsafe (singleton): Unsafe due to singleton suppression.
- Unsafe (singleton) (manual): Unsafe due to singleton suppression but primary suppression carried out manually.
- Unsafe (from manual): manually made unsafe during this session.
- Protected: Cannot be selected as a candidate for secondary cell suppression.
- Secondary: Cell selected for secondary suppression.
- Secondary (from manual): Unsafe due to secondary suppression after primary suppressions carried out manually.
- Zero: Value is zero and cannot be suppressed.
- Empty: No records contributed to this cell and the cell cannot be suppressed.

Change Status

The second pane ('Change Status') on the right will allow the user to change the cell–status.

- Set to Safe: A cell, which has failed the safety rules is here declared safe by the user.

- Set to Unsafe: A cell, which has passed the safety rules is here declared to be unsafe by the user.

- Set to Protected: A safe cell is set so that it cannot be selected for secondary suppression.

- Use 'a priori' information (see below).

A Priori Info

This option is an *a priori* option to be mainly used for microdata which allows the user to feed τ-Argus a list of cells where the status of the standard rules can be overruled i.e. the status of the cells is already specified. The associated file specifying this information is free format. The format will be:

Code of first spanning variable, Code of second spanning variable, Status of cell (u = unsafe, p = protected (not to be suppressed), s = safe).

```
    Nr, 4, u
    Zd, 6, p
```

Recode

The recode button will bring the user to the recoding system. Recoding is a very powerful method of protecting a table. Collapsed cells usually have more contributors and therefore tend to be much safer.

Hierarchical Recoding

This window shows the codes awaiting recoding
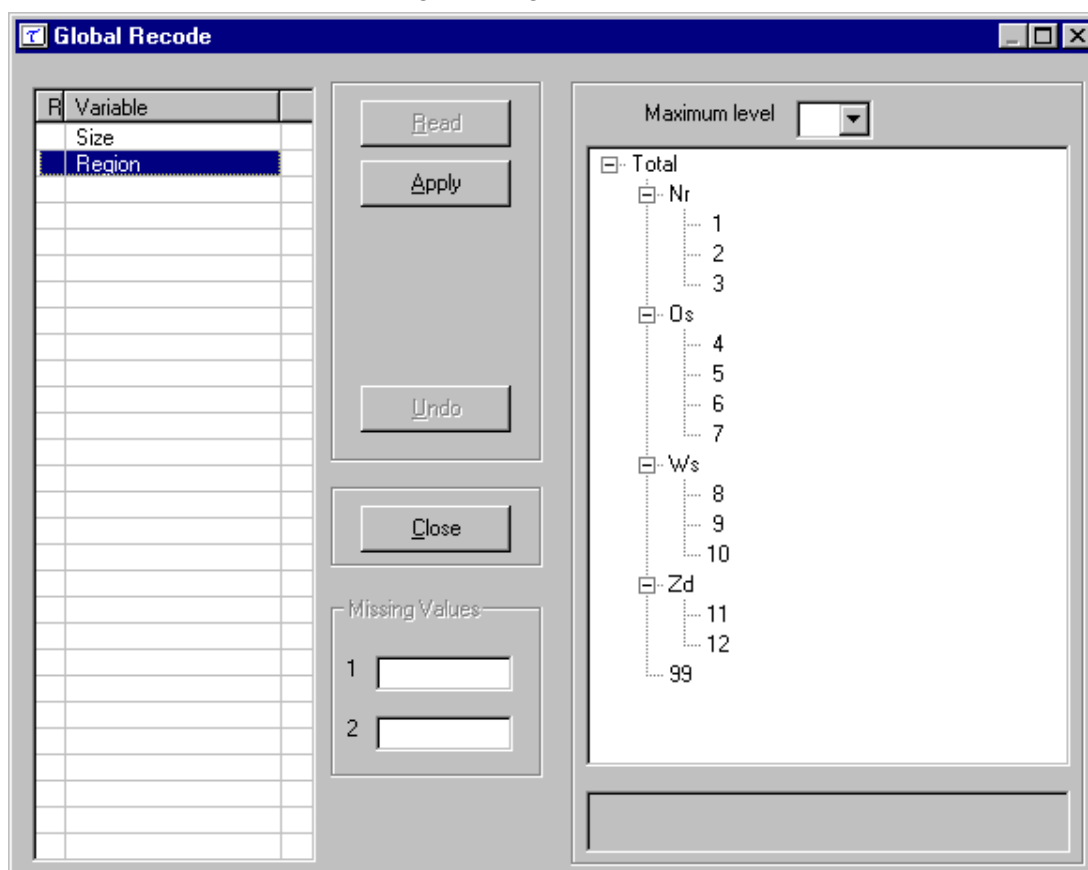
In this example, the 'Region' variable to recode has been selected. The codes are shown in a hierarchical tree. The user can either fold or unfold the branches by clicking on the '+' or '-' boxes which results in showing or omitting codes from the table, or by choosing an overall maximum hierarchical level. (See the following windows for details). Pressing the 'Apply' button followed by 'Close' will actually apply the selected recoding. Press the undo-button – it is now possible to go back to the original recoding scheme. Below this there are two windows, one showing the recode window prior to applying the recoding for 'Region' and the second showing the table following recoding.

This window shows the new hierarchical codes awaiting application
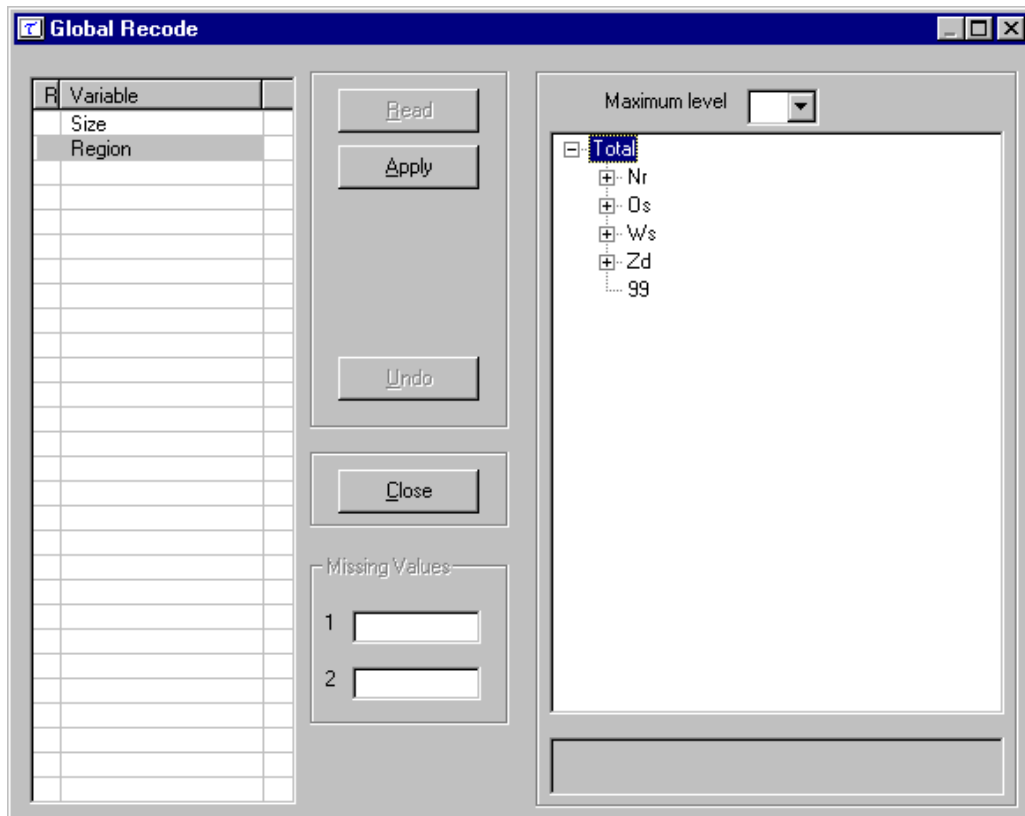
By clicking 'Apply', we obtain this window which shows the table after recoding

	tot	2	4	5	6	7	8	9	99
tot	16,847,647	20	25	2,711,808	2,320,534	2,505,043	2,799,074	6,510,758	385
Nr	4,373,664	5	5	719,049	659,680	688,962	756,529	1,549,049	385
Os	3,703,896	15	5	642,238	515,003	534,147	620,392	1,392,096	-
Ws	4,576,116	-	-	648,972	543,570	663,897	775,132	1,944,545	-
Zd	4,193,971	-	15	701,549	602,281	618,037	647,021	1,625,068	-
99		-	-	-		-	-		-

Cell Information

Value	16,847,647
Status	Safe
Cost	16,847,647
Shadow	16,847,647
# contributions	42723
Top n of shadow	175,677
Holding	141,482
level	135,469
Request	0

Change status
- Set to Safe
- Set to Unsafe
- Set to Protected
- A priori info

Recode

Suppress
- HyperCube
- Modular
- Network
- Optimal

Singleton
Undo Singleton
Suppress
Undo Suppress
Audit

☑ 3 dig. separator
☐ Output View

Select Table
Change View
Table Summary
Write table
Close

In this example the non-hierarchical 'Size' variable has been selected to be recoded. The user can either write the required recodings in the edit box or import them from a previously written file. In the example the line 2:2-6 results that categories 2,3,4,5,and 6 will be recoded into a new category 2, whilst categories 7,8 and 9 will be recoded into the new category 3.

Once the recoding has been applied (both for hierarchical and non hierarchical data) the table can again be displayed. If there are now no cells, which fail the safety rules, the table can be saved as a protected table. However, if there are still a number of unsafe cells, secondary suppression needs to be carried out. This is necessary as the table is not yet safe. If only the cells failing the safety rules are suppressed, other cell values could be obtained by differencing.

Secondary Suppression

The Suppress button is an important button. It will activate the modules for computing the necessary secondary suppressions as described above. There are a number of options here.
- Singleton
- Hypercube
- Modular
- Network
- Optimal

Singleton Suppression

A singleton is a cell with only one contributor. Often such a cell is unsafe, due to a particular sensitivity rule. Singleton protection is only a pre-processing for additional protection of singleton cells. Further details about this approach can be found in the Reference section 4.4.2.

Hypercube

This is also known as the GHMITER method. The approach builds on the fact that a suppressed cell in a simple n-dimensional table without substructure cannot be disclosed exactly if that cell is contained in a pattern of suppressed, nonzero cells, forming the corner points of a hypercube.

Modular

This partial method will break the hierarchical table down to several non-hierarchical tables, protect them and compose a protected table from the smaller tables. As this method uses the optimisation routines, an LP-solver is required: this will be either XPRESS or CPLEX. The routine used can be specified in the Options window, this will be discussed later.

Optimal

This method protects the hierarchical table as a single table without breaking it down into smaller tables. As this method uses the optimisation routines, an LP-solver is required: this will be either XPRESS or CPLEX. The routine used can be specified in the Options window, this will be discussed later.

Network

This is a Network Flow approach for large unstructured 2 dimensional tables or a 2 dimensional table with one hierarchy (the first variable specified).

Choose the suppression method

The radio-buttons at the right lower part of the window allow to select the desired suppression method. Clicking on the Suppress button will then start the process of calculating the secondary suppressions. When this process has finished the protected table will be displayed and also the user will be informed about the number of cells selected for secondary suppression and the time taken to perform the operation. The secondary suppressed cells will be shown in blue.

Table: Size x Region | Var2

	tot	2	4	5	6	7	8	9	99
tot	16,847,647	20	25	2,711,808	2,320,534	2,505,043	2,799,074	6,510,758	385
Nr	4,373,664	5	5	719,049	659,680	688,962	756,529	1,549,049	385
1	1,986,129	5	5	398,062	348,039	354,711	418,778	466,529	-
2	1,809,246	0	-	223,990	221,332	241,913	258,233	863,393	385
3	578,289	-	-	96,997	90,309	92,338	79,518	219,127	
Os	3,703,896	15	5	642,238	515,003	534,147	620,392	1,392,096	-
4	124,336	5	-	36,311	32,132	25,770	18,150	11,968	
5	526,279	-	-	93,589	94,957	110,930	81,799	145,004	
6	2,234,995	10	5	345,803	251,358	251,188	303,377	1,083,254	-
7	818,286	-	-	166,535	136,556	146,259	217,066	151,870	
Ws	4,576,116	-	-	648,972	543,570	663,897	775,132	1,944,545	
8	485,326	-	-	63,767	75,442	87,305	59,953	198,859	
9	3,664,560	-	-	537,911	430,851	515,020	643,762	1,537,016	
10	426,230	-	-	47,294	37,277	61,572	71,417	208,670	
Zd	4,193,971	-	15	701,549	602,281	618,037	647,021	1,625,068	-
11	2,752,743	-	15	488,613	392,395	363,490	402,925	1,105,305	-
12	1,441,228	-	-	212,936	209,886	254,547	244,096	519,763	
99	-	-	-	-	-	-	-		

Cell Information

Value: 16,847,647
Status: Safe
Cost: 16,847,647
Shadow: 16,847,647
contributions: 42723
Top n of shadow: 175,677 / 141,482 / 135,469
Holding level
Request: 0

Change status: Set to Safe | Set to Unsafe | Set to Protected | A priori info

Recode

Suppress: HyperCube | Modular | Network | Optimal | Singleton | Undo Singleton | Suppress | Undo Suppress | Audit

☑ 3 dig. separator ☐ Output View | Select Table | Change View | Table Summary | Write table | Close

Summary Window

By clicking on 'Table Summary', the summary window is obtained. The summary window gives an overview of the cells according to their status.

Freq: The number of cells in each category

rec: The number of observations in each category

Sum Resp: Total cell value in each category

SumCost: The sum of the cost variable. Here it is equal to the response variable.

By clicking on 'OK', we return to the table window.   The table may now be written as an output file in the required format. Any cells which have been selected for suppression will be replaced by 'X'. The safe table can be saved by using the 'Write table' button in this window, or by using 'Output|Save table on the main menu.

Summary for table no: 1

Explan. Var	# Codes
Size	9
Region	18

Status	Freq	# rec	Sum Resp	SumCost
Safe	96	236141	94869102.04	94869102.04
Safe (manual)	0	0	0	0
Unsafe	12	52	12058	12058
Unsafe (request)	0	0	0	0
Unsafe (Freq)	0	0	0	0
Unsafe (Zero cell)	0	0	0	0
Unsafe (Singleton)	0	0	0	0
Unsafe (Singleton) (m...	0	0	0	0
Unsafe (manual)	0	0	0	0
Protected	0	0	0	0
Secondary	11	20145	6204721	6204721
Secondary (fr. man.)	0	0	0	0
Empty (non-struct)	0	0	0	0
Empty	43	0	0	0
----------	----------	----------	----------	----------
Total	162	256338	101085881.04	101085881.04

Respons Var — Resp. Var: Var2

Shadow Var

Cost Var

Protected by Hypercube

OK

3.2.2 Save the safe table

When the table is safe it may be written to the hard disk of the computer. The user has four options:

Save Table

Format
- ◉ CSV-format
- ○ CSV for pivot table   ☐ Add Status
- ○ Code-value   ☐ Suppress empty cells   ☐ Add Status
- ○ Intermediate format   ☐ Status only

Save   Cancel

1. As a CSV file. This Comma separated file can easily be read into Excel. Please note that τ-ARGUS uses the ',' as the field-separator in this CSV-file. This might influence opening the CSV file in Excel. A solution for this is to change the settings in the Windows control-panel. This is a typical tabular output maintaining the appearance of the table in τ-ARGUS.

2. A CSV-file for a pivot table. This offers the opportunity to make use of the facilities of pivot table in Excel. The status of each cell can be added here as an option (Safe, Unsafe or Protected for example). The information for each cell is displayed on a single line unlike standard csv format.

3. A text file in the format code-value, this is separated by commas. Here, the cell status is again an option. Also empty cells can be suppressed from the output file if required. The information

for each cell is displayed on a single line similar to the CSV file for a pivot table.

4. A file in intermediate format for possible input into another program. This contains protection levels and external bounds for each cell. This table could even be read back into τ-ARGUS.

Finally, a report will be generated to a user specified directory. This report will also be displayed on the screen when the table has been written. It will contain details such as table structure, safety rules (and number of cells failing), secondary suppression method (and number of cell failing) and details of any recodes. An example is shown in the Reference section 4.5.2. As this is an HTML-file it can be viewed easily later.

4. Reference Section - description of the Menu Items

Chapter 3 gave a brief overview of the most frequently used options within τ-ARGUS. In this section a more detailed description of the program by menu-item is presented. The information in this section is the same as the information shown when the help-facility of τ-ARGUS is invoked. Note that all tables presented here are magnitude tables.
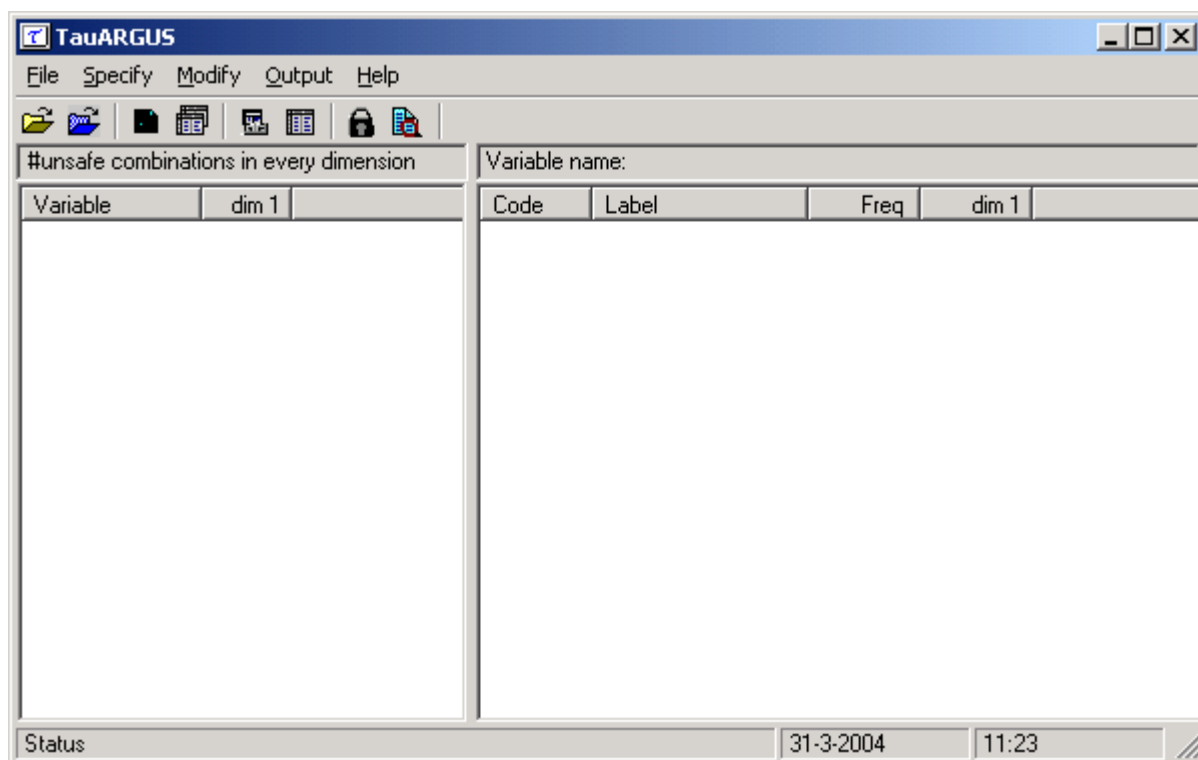
4.1 Main Window

There are five menu headings:

File	Under **File** either a microdata file or tabular data file can be opened; in addition there is the option to open a Batch process file and to Exit.
Specify	**Specify** allows the metadata to be entered or edited as well as letting the user specify the tables of particular interest along with primary suppression rules.
Modify	Under **Modify,** the table can be selected, viewed and any secondary suppressions carried out. Also secondary suppression for linked tables can be performed.
Output	**Output** allows the suppressed table to be saved. In addition there is also view report and write batchfile
Help	Finally, there is a **Help** menu, with contents, options and about the product.

.

Below is a list of the menu items which are shown under each of the menu headings. As some of the items are context specific they will not all be always available.

Overview of the menu-items

File	**Specify**	**Modify**	**Output**	**Help**
Open Microdata	**Metafile**	**Select Table**	**Save table**	**Contents**
Open Table	**Specify Tables**	**ViewTable**	**View Report**	**Options**
Open Batch Process		**Linked Tables**	**Write Batch File**	**About**
Exit				

These menu items will be explained in detail in the following sections.

4.2 The File menu

τ-Argus can read data in two ways. The first of these is microdata (both fixed and free format), which is explained in section 4.2.1. The second is input and treatment of a pre-formed tabulated data and is dealt with in section 4.2.2. Only one of these options can be used at one time, a table and a set of microdata cannot be read in τ-Argus simultaneously. τ-ARGUS can also be used in batch, see section 4.2.3

4.2.1 File | Open Microdata

The File|Open microdata menu allows the user to specify the microdata file (both fixed and free format) and the metadata file
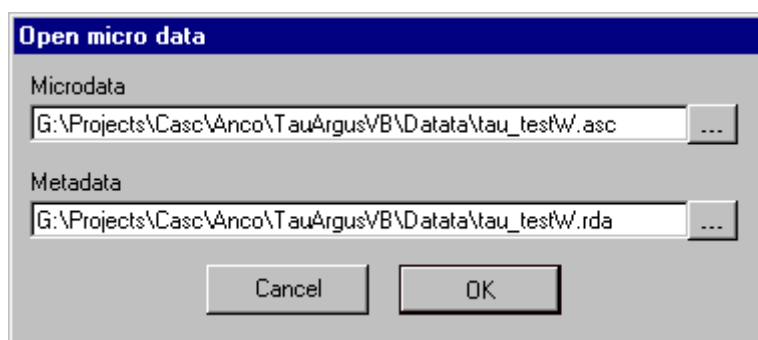
In this dialog box the user can select the microdata-file and the corresponding metadata file

By default the microdata-file has extension .asc and the metafile .rda. (Note, the user may use any file extension, but is advised to use default names).

- When the user clicks on ⌐…⌐ they get an open file dialog box. This box enables searching for the required files. Other file-types can be chosen when clicking on the file-types listbox. When the user has selected the microdata file a suggestion for the metafile (with the same name but with the extension .rda) is given but only when this file exists. Note, both files do not have to have the same name.

The metafile describes the variables in the microdata file, both the record layout and some additional information necessary to perform the SDC-process. Each variable is specified on one main line followed by one or more option lines.

1. The first line gives the name of the variable followed by the starting position for each record, the width of the field and optionally one or two missing value indicators for the record. Missing values are no longer required in τ-ARGUS.
2. The following lines explain specific characteristics of the variable:

• <RECODEABLE>	This variable can be recoded and used as an explanatory variable in a table
• <CODELIST>	This explanatory (or spanning) variable can have an associated codelist which gives labels to the codes for this particular variable. The name of the codelist file follows this <CODELIST> command. The default extension is .CDL. See below rda file for an example of a codelist file.
• <NUMERIC>	This numeric variable can be used as cell-item.
• <DECIMALS>	The number of decimal places specified for this variable
• <WEIGHT>	This variable contains the weighting scheme
• <HIERARCHICAL>	This variable is hierarchical. The codings are structured so that there is a top code such as Region (N,S,E,W) and within each of these are smaller more specific areas (and possibly sub-areas). Tables may be viewed at different levels of hierarchy.
• <HIERLEVELS>	The hierarchy is derived from the digits of the codes itself. The specification is followed by a list of integers denoting the width of each level. The sum of these integers should be the width of the total code. An example is shown beneath the rda file below.
• <HIERCODELIST>	The name of the file describing the hierarchical structure. Default extension .HRC. An example is shown following the rda file.
• <HIERLEADSTRING>	The string/character that is used to indicate the depth of a code in the hierarchy. An example is shown below.
• <REQUEST>	This variable contains the status denoting whether or not a respondent asked for protection
• <HOLDING>	This variable contains the indication whether a group of records belong to the same group/holding

An example of a metafile i.e. an rda file is shown here for a fixed format file. An example for a free-format meta datafile is given at the end of this section.

```
YEAR 1  2 99
  <RECODEABLE>
IndustryCode 4  5 99999
  <RECODEABLE>
  <HIERARCHICAL>
  <HIERLEVELS>  3 1 1 0 0
Size 9  2 99
  <RECODEABLE>
Region 12  2 99
  <RECODEABLE>
  <CODELIST> Region.cdl
  <HIERCODELIST>  Region2.hrc
  <HIERLEADSTRING> @
  <HIERARCHICAL>
```

```
Wgt 14  4 9999
   <NUMERIC>
   <DECIMALS>  1
   <WEIGHT>
Var1 19  9 999999999
   <NUMERIC>
Var2 28  10 9999999999
   <NUMERIC>
   <DECIMALS>  2
......
......
```

Explanation of the file and details of the variables

'Year': For this explanatory/spanning variable each record begins on position 1, is 2 characters long and missing values are represented by 99. It is also recodeable implicitly stating that it is an explanatory or spanning variable used to create the tables.

'IndustryCode': For this variable each record begins on position 4 and is 5 characters long. Missing values are represented by 99999. As well as being recodeable this variable is hierarchical and the hierarchy structure is specified. The first 3 characters are in the top hierarchy level, the 4th character in the second level and the 5th character in the lowest level. As 'Industry' is a 5 digit variable there are 5 digits specified for the hierarchical structure. This is the reason for the 2 zeros at the end.

'Size': For this variable each record begins on position 9 and is 2 characters long, and missing values are represented by 99. It is also recodeable.

'Region': For this variable each record begins on position 12 and is 2 characters long. Missing values are represented by 99. An example of a codelist file can be found in region.cdl and of a hierarchical codelist file in region2.hrc. Contents of these files are shown here.

region.cdl

```
 1,Groningen
 2,Friesland
 3,Drenthe
 4,Overijssel
 5,Flevoland
 6,Gelderland
 7,Utrecht
 8,Noord-Holland
 9,Zuid-Holland
10,Zeeland
11,Noord-Brabant
12,Limburg
Nr,North
Os,East
Ws,West
Zd,South
```

For *region2.hrc* the string/character that is used to indicate the depth of a code in the hierarchy (HIERLEADSTRING) is @.

Note that the total code is never specified in these .HRC files.

region2.hrc

```
Nr
@ 1
@ 2
@ 3
Os
@ 4
@ 5
@ 6
@ 7
Ws
@ 8
@ 9
@10
Zd
@11
@12
```

'Wgt': For this variable each record begins on position 14 and is 4 characters in length with missing values represented by 9999. There is 1 decimal place for these values and the variable is defined as a weight.

Two numeric variables are also shown in the above rda file. These numeric variables (not defined as weights) are those to be used as cell items *i.e.* response variables used in creating the table.

'Var1': This variable begins on position 19 and is 9 characters long. Missing values are represented by 999999999 and it is numeric.

'Var2': This variable begins on position 28 and is 10 characters long. Missing values are represented by 9999999999 and it is numeric. This variable has 2 decimal places.

The representation in an rda file for the Request rule and Holding Indicator are shown here for completeness.

Request rule

```
Request 99  1
  <REQUEST>  "1" "2"
```

Here the request indicator is in column 99 and is one character long. Individuals (or companies) wishing to make use of this rule are represented by 1 or 2, other the variables takes the value 0. Two different parameters-sets for the request rule can be specified, the first set will be applied to the companies where the first code has been specified, the second set to the companies with the second code. The request rule is further explained in section 4.3.3.

Holding Indicator

```
entgroup 101 1
  <HOLDING>
```

Here the variable 'entgroup' is in column 101 and is one character long. This variable is to act as the holding indicator (see section 4.3.1 for further explanation). The records of a holding should be grouped together in the input datafile. τ-ARGUS will not search through the whole file to try to find all records for a holding.

Free format mirodata

For a free-format datafile the RDA is a little bit different. Notably the first line specifies the separator used. This indicates to τ-ARGUS that the record description is for a free-format file. And for each variable the starting position is no longer specified, as this is meaningless in a free-format datafile. For the rest there are no differences compared to the fixed format version. The example given above for a fixed format file will now looks as:

```
<SEPARATOR> ","
YEAR    2 99
   <RECODEABLE>
Sbi    5 99999
   <RECODEABLE>
   <HIERARCHICAL>
   <HIERLEVELS>  3 1 1 0 0
GK    2 99
   <RECODEABLE>
Regio   2 99
   <s RECODEABLE>
   <CODELIST>    REGION.CDL
   <HIERARCHICAL>
   <HIERCODELIST>    region2.hrc
   <HIERLEADSTRING>  @
Wgt  4 9999
   <NUMERIC>
   <DECIMALS>  1
   <WEIGHT>
Var1   9 999999999
   <NUMERIC>
Var2   10 9999999999
   <NUMERIC>
   <DECIMALS>  2
……
……
```

4.2.2 File | Open Table

This is the option allowing the input of tabular data into τ-Argus. In this case, an already-constructed table is read in. This is reached by selecting 'Open a Table' on the main window of τ-ARGUS.
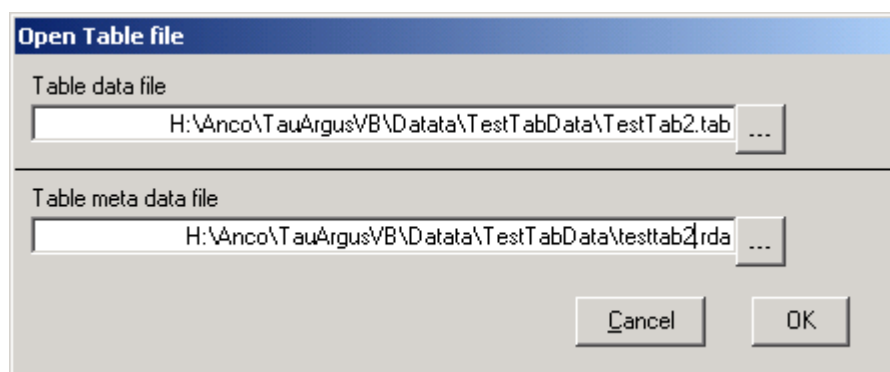
The name of the datafile containing the table to be opened (in the format given below) needs to be specified in the top line. The name of the file containing the metadata is entered on line 2. Later on you will be offered the option of adapting the metadata or even enter the metadata from scratch.

There is a great flexibility with this option as it allows the status, the cell frequency, the top n values and the lower and upper protection levels to be entered for each cell. The more detail is given for each cell to more flexibility τ-ARGUS offers in a later stage to apply sensitivity rules etc.

Here, by clicking 'OK' this allows both re-specification of the metafile under the Specify|Metafile option and the setting safety rules using the 'Specify|Table Metadata' option.

Format (An example of a 2 dimensional table)

This (artificially generated) datafile shows 2 explanatory variables, cell value, cell frequency, the top 3 values in each cell and an indication of whether a cell is safe or unsafe.

```
T,  T, 2940 ,48,  200,200,200 ,u
T,  A,  745 ,12,  200,100,100 ,s
T,  B,  810 ,12,  200,100,100 ,s
T,  C,  685 ,12,  200,100,100 ,s
T,  D,  700 ,12,  200,100,100 ,s
1,  T,  795 ,12,  200,100,100 ,s
1,  A,  350 ,3,   200,100,50  ,s
1,  B,  190 ,3,   100,50,40   ,s
1,  C,  150 ,3,   100,40,10   ,s
1,  D,  115 ,3,   50,40,25    ,s
2,  T,  670 ,12,  200,100,100 ,s
2,  A,  115 ,3,   50,40,25    ,s
2,  B,  340 ,3,   200,100,40  ,s
2,  C,  115 ,3,   50,40,25    ,u
2,  D,  120 ,3,   100,10,10   ,u
3,  T,  785 ,12,  200,100,100 ,s
3,  A,  190 ,3,   100,50,40   ,s
3,  B,  115 ,3,   50,40,25    ,s
3,  C,  325 ,3,   200,100,25  ,s
3,  D,  165 ,3,   100,40,25   ,s
4,  T,  690 ,12,  200,100,100 ,s
4,  A,  100 ,3,   50,25,25    ,s
4,  B,  175 ,3,   100,50,25   ,s
4,  C,  115 ,3,   50,40,25    ,s
4,  D,  310 ,3,   200,100,10  ,s
```

Indication of whether a cell is safe or unsafe is optional and if safety rules are to be applied they will override these indicators. For tables of dimension 3 or higher, additional columns for the explanatory variables would have to be added as well as additional rows to allow for the increased depth of the table.

The next stage is to allow the metafile to be edited.

4.2.3 File | Open Batch Process

This option allows the user to run the commands in batch mode from opening the microdata and metadata to output of the final table(s). A file can be written in a text editor and called from this command.

The possible commands are shown here.

Command	Parameters
OPENMICRODATA	Data file name with microdata
OPENTABLEDATA	File name containing tabular data
OPENMETADATA	Metadata file name
SPECIFYTABLE	"ExpVar1""ExpVar2""ExpVar3"|RespVar|ShadowVar|Costvar, Lambda Shadow and cost variables are optional. If not specified then they equal the Response Variable. If the cost variable is specified either a variable is specified or –1 is chosen for frequency or -2 for unity For lambda the default is 1. (See section 4.3.3 for the explanation for the use of lambda)
CLEAR	Clears all and start a new session.
SAFETYRULE	This command is used for primary suppression. A set of safety rule specifications separated by a "|" Each safety spec starts with "P", "NK" "ZERO", "FREQ", "REQ" and between brackets the parameters **P**: (p,n) with the n optional.(default = 1). So (20,3)➔. p=20% and n=3 **NK: (**n,k**)** **ZERO**: (ZeroSafetyRange) **FREQ**:(MinFreq, FrequencySafetyRange) **REQ**: (Percent1, Percent2, SafetyMargin) All rules can appear several times, The first two **P, NK** are for the individual level; the following two for the holding level, The first FREQ and REQ are at the individual level the second one is for the holding. ZERO, the zero safety range parameter, can be given only once for each safety rule.
READMICRODATA	Just reads the microdata file and calculates the table; no parameters are required
READTABLE	Just reads the tabular inputfile; no parameters are required
APRIORI	This reads an a Priori file The parameters are: Filename, Table number and the separator "Filename", TabNo, Separator
SUPPRESS	This command applies the secondary suppression. The parameters are a solution with a few parameters between brackets **GH**(TabNo, A priori Bounds Percentage) **MOD**(TabNo) **OPT**(TabNo, MaxComputingTime) **NET**(TabNo)
WRITETABLE	(TabNo, P1, P2, FileName) See also section 4.5.1
	P1 (OutputType)	P2 (parameter)

	1. CVS-file	Not used
	2. CSV file for pivot table	1 = AddStatus 0 = not
	3. Code,value file	1 = AddStatus
		2 = suppress empty cells
		3 = both options
		0 = none
	4. Intermediate file	0 = Status only
		1 = also Top-n scores
GOINTERACTIVE	Should be the last command. If omitted the program will stop. If specified τ-ARGUS will go on as an interactive program.	

A typical batch file would look like this: (note that everything after a // will be treated as comment)

```
   //datafile
<OPENMICRODATA> "C:\Program Files\TauARGUS\datatau_testW.asc"
   //metafile
<OPENMETADATA>   "C:\Program Files\TauARGUS\datatau_testW.rda"
   //Exp|resp|shadow|cost −1= unit −2 = freq
<SPECIFYTABLE>    "Size""region"|"var2"|"var3"|"var3"
<SAFETYRULE>      P(20,3)|FREQ(3,30)|ZERO(20)
<SPECIFYTABLE>    "Size""Year"|"var2"|"var3"|"var3"
<SAFETYRULE>      NK(3,70)|FREQ(3,30)|ZERO(20)
<READMICRODATA>
<SUPPRESS>     GH(1,75)
<WRITETABLE>   (1,1,1,"C:\Program Files\TauARGUS\datax1.csv")
<SUPPRESS>     GH(2,75)
<WRITETABLE>   (2,2,1,"C:\Program Files\TauARGUS\datay11.csv")
<SUPPRESS>     MOD(1)
<WRITETABLE>   (1,3,0,"C:\Program Files\TauARGUS\datax20.txt")
<SUPPRESS>     MOD(2)
<WRITETABLE>   (2,4,0,"C:\Program Files\TauARGUS\datay20.tab")
<SUPPRESS>     OPT(1,5)
<WRITETABLE>   (1,1,1,"C:\Program Files\TauARGUS\datax3.csv")
   //<GOINTERACTIVE>
```

The batch file can be used in a real batch environment as well. Just invoke τ-ARGUS with the command

Taupath\TAUARGUS param1 param2

where taupath is the name of the directory where you installed τ-ARGUS, param1 is the name of the above described file with batch commands. Param2 is optional, and is the name of the logfile. If omitted τ-ARGUS will write a logbook in the file LOGBOOK.TXT in the temp-directory. See also section 4.7

4.2.4 File | Exit

Exits from the τ-ARGUS-session.

4.3 The Specify menu

4.3.1 Specify | Metafile [for microdata]

Clicking on 'Specify|metafile' gives the user the opportunity to either edit the metafile already read in or to enter the metafile information directly at the terminal.

In this dialog box all attributes of the variables can be specified. This is a good alternative to editing the rda file outside τ-ARGUS. τ-ARGUS does a moderate checking of the rda-file, but no guarantee can be given for a proper functioning of a manually edited RDA-file. The rda file has been explained in detail in section 4.2.1. Here, editing of a rda file within τ-ARGUS is looked at.

If under File|Open Microdata an *rda* file has been specified, this dialog box shows the contents of this file. If no .rda file has been specified the information can be specified in this dialog box after pushing the New button. As default "newvar" is substituted. Apart from defining a new variable, an existing one can be modified or deleted.

An example of the metafile window is shown here.

In the left top field the file type (fixed or free format) can be specified.

The following attributes for each variable can be specified or edited:
- name of the variables
- its first position in the data file
- its field-length
- the number of decimals.

Furthermore, the kind of variable can be specified or edited (more detail on these can be seen in section 4.2.1):

- *explanatory variable*: This can be used as a spanning variable in the row or column of the table

- *response variable*: This can be used as a cell-item

- *weight variable*: This specifies the weight of the record and is based on the sampling design

---

used.

The following are specialist variable types and have not been previously described. As they are specific to designating safety rules, more detail is given in section 4.3.3.

Holding Indicator
The Holding indicator: sometimes groups of records belong together. So it could be better to apply the confidentiality protection to businesses at a number of levels. This variable is the group identifier. τ-ARGUS expects the records of a group to be together in the input datafile. An example is shown in section 4.3.3.

Request Protection
The Request protection option is used if the Request Rule under 'Specify tables' is to be applied. This variable indicates whether or nor a records asked for protection. This is further explained in section 4.3.3. Additionally the codes specifying whether a respondent asked for asking protection is to be specified; two different codes are possible, corresponding to two different sets of parameters in the sensitivity rule.

Additional Specifications
Other attributes, which may be edited or specified are
   • missing value options, (optional, not required)
   • codelist files
   • hierarchies.
Details on these options have been given in section 4.2.1.
In summary, for codelist the 'automatic' option simply generates the codes from the data. Specifying a codelist, allows the user to supply an additional file (usually .cdl) containing the labels attached to the codes. These labels are used to enhance the information by τ-ARGUS on the screen. In both cases τ-ARGUS will use the codes that it finds in the datafile.
Hierarchies can either be derived from the digits in the codes or from a file (usually .hrc)

The Rda file

Here is an example of a rda file for microdata. This has already been shown in section 4.2.1 and is shown here for completeness. (Note, the dots at the bottom just means that here a shortened version of the file is presented.)

```
YEAR 1   2 99
   <RECODEABLE>
IndustryCode 4   5 99999
   <RECODEABLE>
   <HIERARCHICAL>
   <HIERLEVELS>   3 1 1 0 0
Size 9   2 99
   <RECODEABLE>
Region 12   2 99
   <RECODEABLE>
   <CODELIST> Region.cdl
   <HIERCODELIST>   Region2.hrc
   <HIERLEADSTRING> @
   <HIERARCHICAL>
Wgt 14   4 9999
   <NUMERIC>
   <DECIMALS>   1
   <WEIGHT>
Var1 19   9 999999999
   <NUMERIC>
Var2 28   10 9999999999
   <NUMERIC>
   <DECIMALS>   2
```

```
......
......
```

4.3.2 Specify | Metafile [for tabular data]

When a tabular datafile has been selected, the metadata window will have a different form. Clicking on 'Specify|metafile' gives the opportunity to either edit the metafile already read in or to enter the metafile information directly at the computer. Below is displayed the 'Specify metafile' window for tabular input data.

Above the list of variables the separator used to separate the variables in the datafile can be specified.

Here, the variables can be specified or edited as required.

The options are:

- *'Explanatory Variable'* – The spanning variables used to produce the table.

- *'Response Variable'*– The variable used to calculate the cell total.

- *'Shadow variable'*– The variable is used as a shadow variable.

- *'Cost variable'*– The variable is used as the cost-variable.

- *'Lower prot .Level'* – The lower protection level

- *'Upper prot. Level'* – The upper protection level

- *'Frequency'* – This indicates the number of observations making up the cell total. If there is no frequency variable each cell is assumed to consist of a single observation.

- *'topN variable'* – This shows if this variable is defined as one of the top N contributors to the cell. The pre-defined value for TopN is 1. The first variable declared as 'topN' will contain the largest values in each cell, the second variable so declared will contain the second largest values etc.

- *'Status Indicator'* – allows a variable in the left-hand pane to be declared as a Status Indicator. Typically cells can be declared as Safe, Unsafe or Protected.

For explanatory variables the code for the total has to be specified. Either the user also provides the values for the totals himself, or he asks τ-ARGUS to compute these totals; in either case,τ-ARGUS needs these totals as they play an important role is the structure of a table and also are important for the suppression models.

The remaining options for these codelists and the hierarchies are the same as for microdata.

The rda file for the above window is shown here.

```
<SEPARATOR>     ",”
expvar1
    <RECODABLE>
    <TOTCODE> T
expvar2
    <RECODABLE>
    <TOTCODE> T
respvar
   <NUMERIC>
freq
   <FREQUENCY>
top1
   <MAXSCORE>
top2
   <MAXSCORE>
top3
   <MAXSCORE>
stat
    <STATUS>
```

4.3.3 Specify | Specify Tables [for microdata]

In this dialog box the user can specify the tables which require protection. In one run of τ-ARGUS more than one table can be specified, but the tables will be protected separately unless they are linked (have at least one variable in common). In that case they can be protected simultaneously if required. In section 4.4.3 the idea of linked tables will be discussed.

Also, the user has to specify the parameters for the dominance rule or p% rule and the minimum number of contributors in a cell, etc. At present τ-ARGUS allows up to 6-dimensional tables, but due to the capacities of the LP-solver used (either Xpress or Cplex depending on the user's license) and the complexity of the optimisations involved, tables of this complexity can only be protected by the hypercube method (see section 2.7 in the Theory chapter).

Below is a typical window obtained when specifying tables with the dominance rule applied.

In section 4.3.1 details of variable definitions in the metafile were explained. Now consider how the variables defined in the metafile are used to create a table along with an associated safety rule.

The explanatory (or spanning) variables
On the left is the listbox with the explanatory variables.
When the user clicks on '>' or '<' the selected variable is transported to the next box. From the left box with explanatory variables the user can select the variables that will be used as the spanning variables in the row or the column of the table.

Cell items

Here, is a list of variables that can be used as response, shadow or cost variables in the disclosure control. By pressing the '>' or '<' they can be transferred to or from the windows on the right.

The response variable

From the list of cell items the user can select a variable as a response variable. This is the variable for which the table to be protected is calculated. If a number of tables are required for the same explanatory variables then more than one response variable can be entered here. Each response variable is suppressed independently.

The shadow variable

The shadow variable is the variable that is used to apply the safety rule. By default this is the response variable, but it is possible to select another variable. The safety rules are built on the principle of the characteristics of the largest contributors to a cell. If a variable other than the response variable is a better indicator this variable can be used here; e.g. the turnover (a proxy for the size of the enterprise) can be a suitable variable to apply the safety rule, although the table is constructed using another (response) variable.

The cost variable

This variable describes the costs of suppressing each individual cell; these costs are used by the internal workings of the secondary suppression routines. These costs are minimised when the secondary suppressed cells are determined. By default, this is the response variable but two other choices are possible as well as the use of a different response variable.

Use the frequency of the cells as a cost-function: this will minimise the number of records contributing to the cells to be suppressed.

The number of cells to be suppressed is minimised, irrespective of the size of their contributions (Unity option).

A Box-Cox like-transformation can be applied to the individual values of the cost variable before minimisation of the cost function. The Box Cox function used here is x^λ where x is the cost variable and λ is the transformation parameter. For example if $\lambda = 0.5$ a square root transformation is used and if $\lambda = 0$ a log transformation will be applied. Applying this to the unity-choice is rather meaningless.

Weight

If the data file has a sample weight, specified in the metadata, the table can be computed taking these weights into account.

There are 2 options.

If the 'Apply Weights' box only is ticked, the weights are applied to the cell entries as for the simple application of normal sampling weights in a survey. This has nothing to do with Disclosure Control, but creates tables with weighting applied.

If the 'Apply Weights in Safety Rule' is also ticked the safety rules themselves use the weights.

For example if there is a cell with two contributions:

100, weight 4

10, weight 7

The cell value = (4 x 100) + (7 x 10) = 470. Without considering the weights there are only two contributors to the cell 100 and 10. However by taking account of the sampling weights the cell values are approximately 100, 100, 100, 100, 10, 10, 10, 10, 10, 10 and 10. The largest two contributors are now 100 and 100. These are regarded as the largest two values for application of the safety rules. If the weights are not integers, a simple extension is applied.

The safety rule

The concept of safety rules is explained in section 2.1 On the left side of the window the type of rule that can be selected along with the value of the parameters is shown. The possible rules are:

- Dominance Rule

- • P% Rule
- • Request Rule (this rule is described in detail later in this section)

Additionally, the minimum number of contributors may be chosen (in the 'minimum frequency' box). Two dominance rules and two P% rules can be applied to each table. When 2 rules are specified, for a cell to be declared non-disclosive, it must satisfy both rules.

Dominance Rule

This is sometimes referred to as the (n,k) rule where n is the number of contributors to a cell contributing more than k% of the total value of the cell (if the cell is to be defined as unsafe). A popular choice would be to set n equal to 3 and k equal to 75%. An example of the window when specifying a single dominance rule is shown at the start of this section.

P% rule

Here is an example of the window when the P% rule is specified.

The P% rule says that if x_1 can be determined to an accuracy of better than P% of the true value then it is disclosive where x_1 is the largest contributor to a cell.

The rule can be written as:

$$\sum_{i=3}^{c} x_i \geq \frac{p}{100} x_1$$ for the cell to be non-disclosive where c is the total number of contributors to

the cell and the intruder is a respondent in the cell.

It is important to know that when entering this rule in τ-ARGUS the value of N refers to the number of intruders in coalition (who wish to group together to estimate the largest contributor).

A typical example would be that the sum of all reporting units excluding the largest two must be at least 10% of the value of the largest. Therefore, in τ-ARGUS set p=10 and n =1 as there is just one intruder in the coalition, respondent x_2.

For the dominance rule and the *P%*-rule the safety ranges required (as a result of applying the rule) can be derived automatically. The theory gives formulas for the upper limit only, but for the lower limit there is a symmetric range. See e.g. Loeve (2001). (This is referenced in Section 2 (Theory))

Request Rule
This is a special option applicable in certain countries relating to e.g. foreign trade statistics. Here, cells are protected only when the largest contributor represents over (for example) 70% of the total and that contributor asked for protection. Therefore, a variable indicating the request is required.
This option requires an additional variable in the data, with e.g. 0 representing no request for that particular business, and 1 representing a request where the particular cell value is > x% of the cell total. In fact there is an option for two different thresholds. The min freq is interpreted such that if a cell has at least one request and the cell-freq is below the freq-threshold, that cell is considered to be unsafe as well. Even if the request is not the largest one. The idea is that in that case a large non requesting contributor could reveal the smaller requesting contributor.

In the example there is a single request threshold (70%) and minimum frequency equal to 3.

Minimum Frequency
If this box is checked, a rule controlling the minimum number of contributors to a cell will be specified. If the number of contributors is less than this value, the cell is considered unsafe.

Freq

Here, the minimum number of contributors can be stated. This is sometimes known as the threshold rule. It is also possible to specify no safety rule apart from a minimum frequency value.

Range

As described above, for the dominance rule and the *P%*-rule safety ranges can be derived automatically. However, the theory does not provide any safety range for the minimum frequency rule. Therefore, the user must provide a safety-range percentage required to allow secondary suppressions to be carried out. For example, if this value was set to equal 30%, it would mean an attacker would not be able to calculate an interval for this cell to within 30% of the actual value when looking at the safe output. Following this, the secondary suppressions may be carried out.

Manual Safety Range

When a cell is set manually unsafe (an option to discussed later), τ-ARGUS cannot calculate safety-ranges itself. Therefore, the user must supply a safety-percentage for this option for the same reasons as in the above section, to allow secondary suppressions to be applied.

Zero Unsafe

If all contributions to a cell are zero, the cell value will be zero too. Applying sensitivity rules here has some problems. Is the sum of the largest 3 zeros larger than zero? Nevertheless all contributions to this cell can be easily disclosed. If cells with total contributions of zero are to be regarded as unsafe, this box has to be checked. A manual safety range will also be required, not as a percentage but as a value at the level of the cell-item.

Holding Indicator
This section on the Holding Indicator is best read after section 4.4.2

In some countries, confidentiality protection is applied to businesses at different levels. For example, as in the U.K. a number of 'reporting units' (the lower level of unit) within a cell might belong to an 'enterprise group' (higher level). The level at which the confidentiality rule is applied clearly matters. The holding indicator allows such groupings to be defined and used in one or more of the safety rules.

This is now illustrated with an example looking at both the P% rule and the threshold rule at the same time.

Consider the following dataset

Cell Ref	Cell Ref	Cell value (reporting unit)	Enterprise group
800	20	599	1
800	20	344	1
800	20	244	1
800	30	355	1
800	20	644	2
800	30	433	2
800	30	323	3
800	30	343	3
900	20	23	4
900	20	43	5
900	20	34	5
900	20	53	5
900	30	700	6
900	30	200	6
900	30	60	7
900	30	40	8
900	30	10	9

Assume the following safety rules
Threshold rule: At least 3 enterprise groups (higher level units) in a cell

---

τ-Argus (3.0) user manual

P% rule: The sum of all the reporting units (lower level units) excluding the largest 2 must be at least 10% of the value of the largest.

There are 4 cells in the table along with the margins. The cell we are interested in here is *Cellref 900,30*: 5 reporting units, 4 enterprise groups

At the reporting unit the values are 700,200,60,40,10

At the enterprise group the values are 900,60,40,10

This rule has been designed so that when the P% rule is applied to this cell:
(a) With reporting units the cell is safe. 10+60+40 = 110. This is greater that 10% of the largest value (70) so the cell is safe.
(b) With enterprise groups the cell is unsafe. 40+10 = 50. This is less than 10% of the largest value (90) so the cell is unsafe.

Apply the threshold rule to the enterprise groups (Hold. =3) and P% rule to the reporting units.

Once again a safety range percentage is required.

The output from the application of this rule is shown below. Two cells fail the threshold rule with the holding rule applied.
The threshold rule has been applied correctly using the holding indicator as the correct cells are safe (that would be unsafe if the holding indicator was not being used).

After all the options have been selected compute the table

When all the necessary information has been given, click '∨' to transport all the specified parameters to the 'listwindow' on the bottom. As many tables as required can be specified but as the size of the memory of a computer is restricted it is not advisable to select too many tables. To modify an already made table press the '^' button.
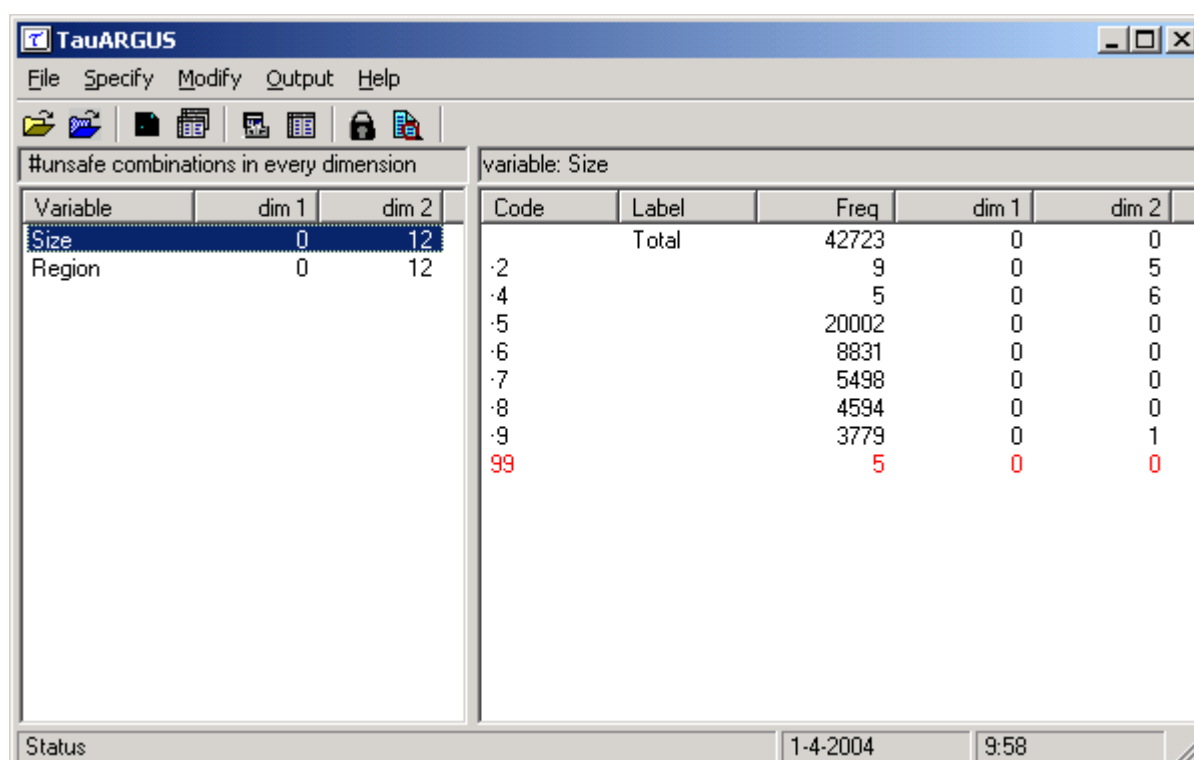
Click on 'Compute Tables' to compute the tables.

When the table(s) have been computed, the main-window of τ-ARGUS will be displayed again with an overview of all the unsafe cells per variable for every table calculated. An example is shown here for the *Size* by *Region* table. Firstly, the Size dimension is looked at and then secondly the Region dimension.

The window (underneath the main menu for τ-ARGUS) shows the number of unsafe combinations per variable. For example, there are no unsafe cells in dimension one for either variable. (*i.e.* the one-way marginal total for different values of 'Size' and 'Region' are all not disclosive).

Size

There are however 12 unsafe cells in the 2 way table *Size* by *Region* as can be seen by the right hand window which gives the equivalent information for each level of the variable indicated on the left. There are 5 unsafe cells where Size = 2, 6 unsafe cells where Size = 4 and a single unsafe cell where Size = 9.

Region

The 12 unsafe cells when looked at by Region show that two of these are 'North' subtotal cells. Within the 'North' region, 2 cells in Groningen are disclosive.

Two 'East' subtotal cells are also unsafe. Within the 'East' region 2 cells in 'Overijssel' are unsafe along with 2 cells in 'Gelderland'

Finally there is 1 unsafe cell for the 'South' subtotal and within this region there is 1 unsafe cell for 'Noord-Brabant'

It should be noted that more than one response variable can be specified. This will produce tables for each of the Response variables using the Spanning variables specified.

4.3.4 Specify | Specify tables [for tabular data]

When the 'Specify|Metafile' option is followed the 'Specify|Table metadata' option is also available and the window is displayed here. This will allow the application of safety rules such as the Dominance Rule and the P% rule. Section 4.3.3 (specifying tables from microdata) will explain these safety rules and other options in detail.

On the left side of the window, the type of rule can be selected along with the value of the parameters. These are the dominance rule and P% rule. Additionally, the minimum number of contributors can be chosen (threshold rule), via ticking and filling-in the minimum frequency box. Note that if the status has been specified, these will prevail.

When all the options have been completed, pressing the 'OK' button will invoke τ-ARGUS to actually compute the tables requested. Now the process of disclosure control can begin.

4.4 The Modify menu

4.4.1 Modify | Select Table

This dialog box enables the user to select the table they want to see. If the user has specified only one table, this table will be selected automatically and this option cannot be accessed. In the example window shown here the first table is a 2 dimensional table (Size x Region) followed by a 3 dimensional table (Size x Region x IndustryCode). Select the table to be processed and press the OK-button.
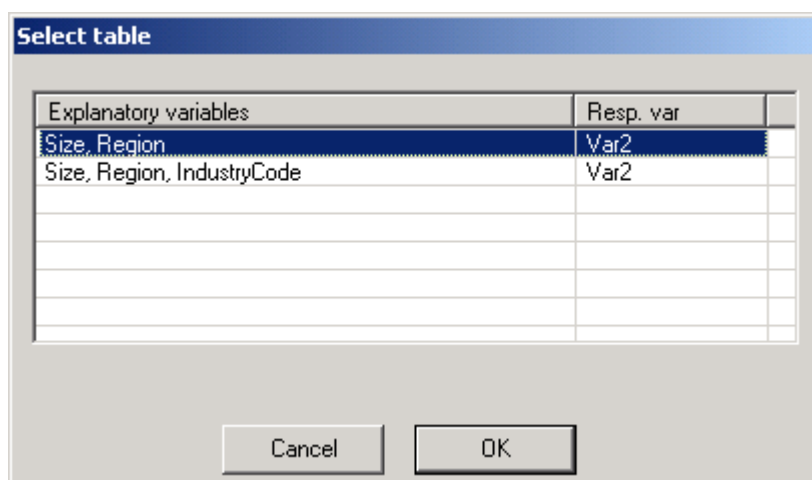
4.4.2 Modify | View Table

This section is divided in four parts: a general description of the 'View Table' screen, global recoding, the secondary cell suppression and some other options

4.4.2.1 The View table screen

This window shows the table selected with Modify|View Table. On the left side the table itself is shown in a spreadsheet view. Safe cells are black, unsafe cells (those failing the primary suppression rule) are red. In this example there are 12 unsafe cells and by viewing the table the user can now see the actual cells that are unsafe.

Any secondary suppressed cells are shown in blue (there are none at this stage, in this example) and empty cells have a hyphen (-). The two check-boxes on the left-bottom give some control over the layout.

• If the 3-digit separator box is checked, the window will show the cell-values will be shown, using the 3 digits separator to give a more readable format.

• The Output view shows the table, with all the suppressed cells replaced by an 'X'; this is how the safe table will be published, but without the colours distinguishing between primary and secondary suppressions.

Table: Size x Region | Var2

	tot	2	4	5	6	7	8	9	99
tot	16,847,647	20	25	2,711,808	2,320,534	2,505,043	2,799,074	6,510,758	385
Nr	4,373,664	5	5	719,049	659,680	688,962	756,529	1,549,049	385
1	1,986,129	5	5	398,062	348,039	354,711	418,778	466,529	-
2	1,809,246	0	-	223,990	221,332	241,913	258,233	863,393	385
3	578,289	-	-	96,997	90,309	92,338	79,518	219,127	-
Os	3,703,896	15	5	642,238	515,003	534,147	620,392	1,392,096	-
4	124,336	5	-	36,311	32,132	25,770	18,150	11,968	-
5	526,279	-	-	93,589	94,957	110,930	81,799	145,004	-
6	2,234,995	10	5	345,803	251,358	251,188	303,377	1,083,254	-
7	818,286	-	-	166,535	136,556	146,259	217,066	151,870	-
Ws	4,576,116	-	-	648,972	543,570	663,897	775,132	1,944,545	-
8	485,326	-	-	63,767	75,442	87,305	59,953	198,859	-
9	3,664,560	-	-	537,911	430,851	515,020	643,762	1,537,016	-
10	426,230	-	-	47,294	37,277	61,572	71,417	208,670	-
Zd	4,193,971	-	15	701,549	602,281	618,037	647,021	1,625,068	-
11	2,752,743	-	15	488,613	392,395	363,490	402,925	1,105,305	-
12	1,441,228	-	-	212,936	209,886	254,547	244,096	519,763	-
99	-	-	-		-		-		-

Cell Information

Value 16,847,647
Status Safe
Cost 16,847,647
Shadow 16,847,647
contributions 42723
Top n of shadow 175,677
☐ Holding level 141,482 / 135,469
Request 0

Change status: Set to Safe | Set to Unsafe | A priori info | Set to Protected

Recode

Suppress: ○ HyperCube | ○ Modular | ○ Network | ○ Optimal | Singleton | Undo Singleton | Suppress | Undo Suppress | Audit

☑ 3 dig. separator ☐ Output View | Select Table | Change View | Table Summary | Write table | Close

For some windows, the complete table cannot be seen on the screen. In these cases there will be scrollbars at the bottom and the right of the table above, which can be used to display the unseen columns.

Example of a 3D table

Variables in 3D tables are displayed at the top left of the typical 2 dimensional table. The arrow at the right of the box allows selection of the required level of this variable. The table shown will be the 2 dimensional table for the specified level(s) of the variables displayed at the top of the table.

Table: Size x Region x Year | Var2

Year: tot

	tot	2	4	5	6	7	8	9	99
tot	16,847,647	20	25	2,711,808	2,320,534	2,505,043	2,799,074	6,510,758	385
Nr	4,373,664	5	5	719,049	659,680	688,962	756,529	1,549,049	385
1	1,986,129	5	5	398,062	348,039	354,711	418,778	466,529	-
2	1,809,246	0	-	223,990	221,332	241,913	258,233	863,393	385
3	578,289	-	-	96,997	90,309	92,338	79,518	219,127	-
Os	3,703,896	15	5	642,238	515,003	534,147	620,392	1,392,096	-
4	124,336	5	-	36,311	32,132	25,770	18,150	11,968	-
5	526,279	-	-	93,589	94,957	110,930	81,799	145,004	-
6	2,234,995	10	5	345,803	251,358	251,188	303,377	1,083,254	-
7	818,286	-	-	166,535	136,556	146,259	217,066	151,870	-
Ws	4,576,116	-	-	648,972	543,570	663,897	775,132	1,944,545	-
8	485,326	-	-	63,767	75,442	87,305	59,953	198,859	-
9	3,664,560	-	-	537,911	430,851	515,020	643,762	1,537,016	-
10	426,230	-	-	47,294	37,277	61,572	71,417	208,670	-
Zd	4,193,971	-	15	701,549	602,281	618,037	647,021	1,625,068	-
11	2,752,743	-	15	488,613	392,395	363,490	402,925	1,105,305	-
12	1,441,228	-	-	212,936	209,886	254,547	244,096	519,763	-
99				-	-	-	-	-	-

Cell Information
Value: 16,847,647
Status: Safe
Cost: 16,847,647
Shadow: 16,847,647
contributions: 42723
Top n of shadow: 175,677 / 141,482 / 135,469
Holding level
Request: 0

Change status: Set to Safe | Set to Unsafe | A priori info | Set to Protected

Recode

Suppress: ⊙ HyperCube | ○ Modular | ○ Network | ○ Optimal | Singleton | Undo Singleton | Suppress | Undo Suppress | Audit

☑ 3 dig. separator ☐ Output View | Select Table | Change View | Table Summary | Write table | Close

Additional information in the View Table window

Clicking on a cell in the main body of the table makes information about this cell visible in the **Cell Information pane.**

Here, the following information can be seen :
1. the cell-value
2. the cell status
3. the total cost variable value for the cell
4. the total of the shadow variables for the cell
5. the number of contributors to a cell
6. the values of the shadow variable for the largest contributors.

Information about the Holding level and the Request protection variable are also displayed here.

The status of the cell can be:
- Safe: Does not violate the safety rule
- Safe (from manual): manually made safe during this session
- Unsafe: According to the safety rule
- Unsafe (request): Unsafe according to the Request rule.
- Unsafe (frequency): Unsafe according to the minimum frequency rule.
- Unsafe (singleton): Unsafe due to singleton suppression (see 'Secondary suppressions' below)
- Unsafe (singleton) (manual): Unsafe due to singleton suppression but primary suppression carried out manually (see 'Change Status' and 'Secondary suppressions' below).
- Unsafe (from manual): manually made unsafe during this session (see 'Change Status' below).
- Protected: Cannot be selected as a candidate for secondary cell suppression (see 'Change Status'

below).
- Secondary: Cell selected for secondary suppression.
- Secondary (from manual): Unsafe due to secondary suppression after primary suppressions carried out manually (see 'Change Status' and 'Secondary suppressions' below).
- Zero: Value is zero and cannot be suppressed.
- Empty: No records contributed to this cell and the cell cannot be suppressed.

Change Status

The second pane ('Change Status') on the right will allow the user to change the cell–status.

- Set to Safe: A cell, which has failed the safety rules is here declared safe by the user.

- Set to Unsafe: A cell, which has passed the safety rules is here declared to be unsafe by the user.

- Set to Protected: A safe cell is set so that it cannot be selected for secondary suppression.

A priori info

This option is to be mainly used for microdata. This allows τ-ARGUS to feed a list of cells where the status of the standard rules can be overruled i.e. the status of the cells is specified from the file. It is free format. The format is:

Code of first spanning variable, Code of second spanning variable, Status of cell (u = unsafe, p = protected (not to be suppressed), s = safe).

```
Nr, 4, u
Zd, 6, p
```

4.4.2.2 Global recoding

The recode button will open the recoding options. Recoding is a very powerful method of protecting a table. Collapsed cells tend to have more contributors and therefore tend to be much safer.

Recoding a non-hierarchical variable

There is a clear difference in recoding a hierarchical variable compared to a non-hierarchical variable.

In the non-hierarchical case the user can specify a global recoding manually. Either enter the recoding described below manually or read it from a file. The default extension for this file is .GRC.

There are some standards about how to specify a recode scheme. All codelists are treated as alphanumeric codes. This means that codelists are not restricted to numerical codes only. However, this also implies that the codes '01' and ' 1' are considered different codes and also 'aaa' and 'AAA' are different. In a recoding scheme the user can specify individual codes separated by a comma (,) or ranges of codes separated by a hyphen (-). The range is determined by treating the codes as strings and using the standard string comparison. E.g. `0111`< `11` as the `0` precedes the `1` and `ZZ`< `a` as the uppercase `Z` precedes the lowercase `a`. Special attention should be paid when a range is given without a left or right value. This means every code less or greater than the given code. In the first example, the new category 1 will contain all the codes less than or equal to 49 and code 4 will contain everything larger than or equal to 150.
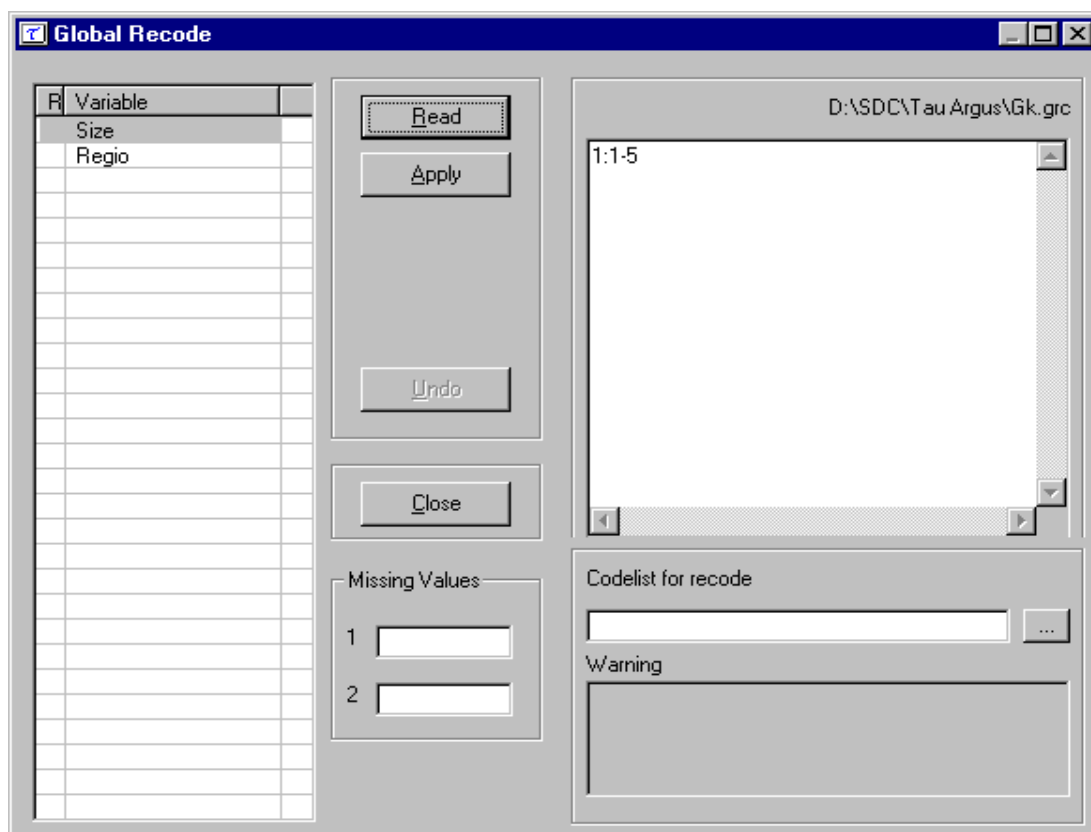
Example:
for a variable with the categories 1,…,182 a possible recode is then:

```
1:        - 49
2:   50  - 99
3:   100 – 149
4:   150 –
```

for a variable with the categories 01 till 10 a possible recode is:

```
1: 01 , 02
2: 03 , 04
3: 05 – 07
4: 08 , 09 , 10
```

An important point is not to forget the colon (:) if it is forgotten, the recode will not work.
Recoding 3: 05,06,07 can be shortened to 3: 05-07.

Additionally, changing the coding for the missing values can be performed by entering these codes in the relevant textboxes.
Also a new codelist with the labels for the new coding scheme can be specified. This is entered by means of a codelist file. An example is shown here. (note, there are no colons is this file)

```
1,Groningen
2,Friesland
3,Drenthe
4,Overijssel
5,Flevoland
6,Gelderland
7,Utrecht
```

```
8,Noord-Holland
9,Zuid-Holland
10,Zeeland
11,Noord-Brabant
12,Limburg
Nr,North
Os,East
Ws,West
Zd,South
```

Pressing the 'Apply' button will actually restructure the table. If required, recoding can easily be undone by pressing 'undo recoding'. The window will return to the originally coding structure. If there is any error in the recoding such as certain codes not being found when pressing the 'Apply' button, an error message will be shown at the bottom of the screen. Alternatively, a warning could be issued; *e.g.* if the user did not recode all original codes, τ-ARGUS will inform the user. This may have been the intention of the user, therefore the program allows it. In the above example a τ-ARGUS message informs the user that 4 codes have not been changed.

Once the 'Close' button has been pressed, τ-ARGUS will present the table with the recoding applied.

Recoding a hierarchical variable
In the hierarchical case the code scheme is typically a tree. To global recode a hierarchical variable
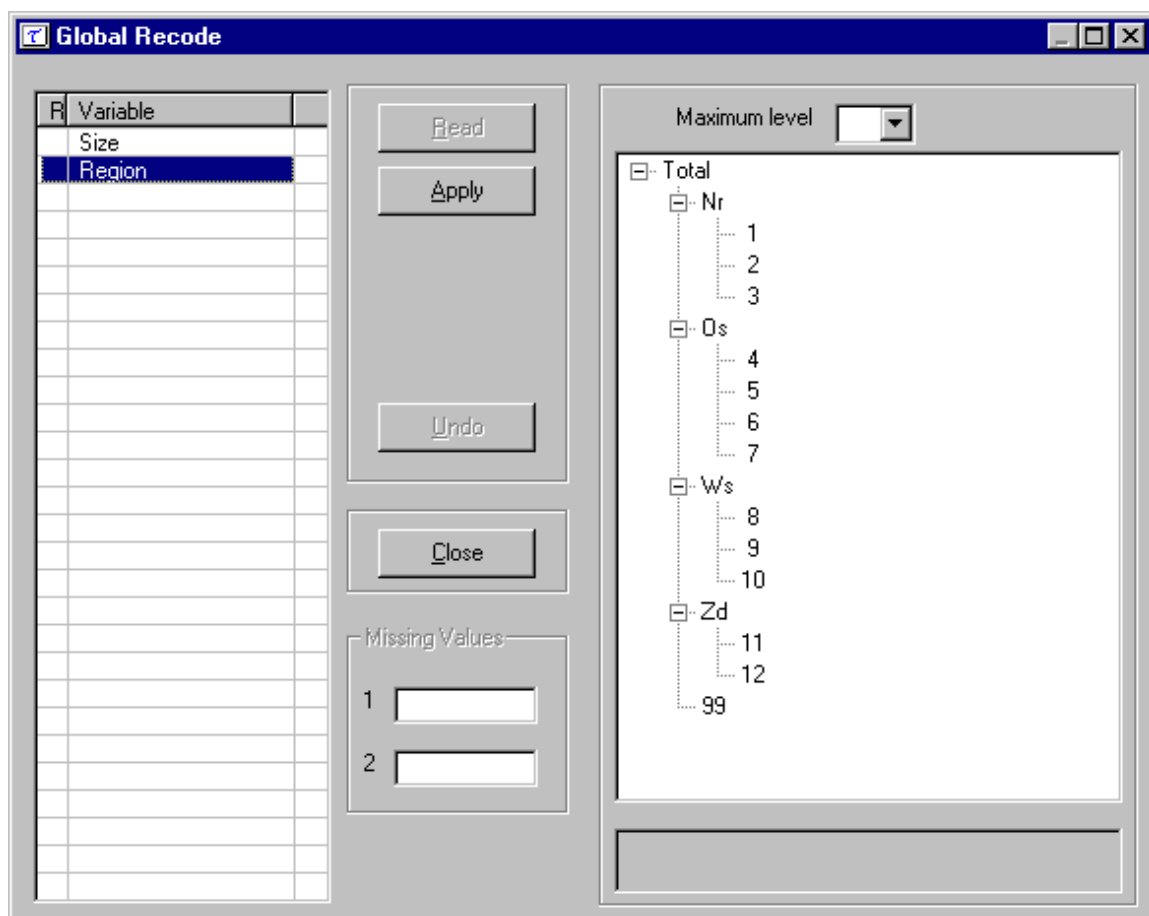
requires a user to manipulate a tree structure. The standard Windows tree view is used to present a hierarchical code. Certain parts of a tree can be folded and unfolded with the standard Windows actions (clicking on '+' and '-').

The maximum level box at the top of the screen offers the opportunity to fold and unfold the tree to a certain level.
Additionally, the user can change the coding for the missing values by entering these codes in the relevant textboxes.

Pressing the 'Apply' button will actually restructure the table. If required, a recoding may always be undone.

4.4.2.3 Secondary suppression

The actions in the suppress pane in the table window (after selecting 'modify|table') are now looked at.
With suppress the table can be protected by causing additional cells to be suppressed. This is necessary to make a safe table.

Suppression Options

There are a number of suppression options, which can be seen on the bottom right hand side of the window.

- Hypercube
- Modular
- Network
- Optimal
- Singleton

Hypercube

This is also known as the GHMITER method. The approach builds on the fact that a suppressed cell in a simple n-dimensional table without substructure cannot be disclosed exactly if that cell is contained in a pattern of suppressed, nonzero cells, forming the corner points of a hypercube.

Selecting the hypercube method will lead to the following window being showed by τ-ARGUS. The user can change these options if required. The ratio parameter for the hypercube approach is set by the software. For detailed information on the hypercube see section 2.7.
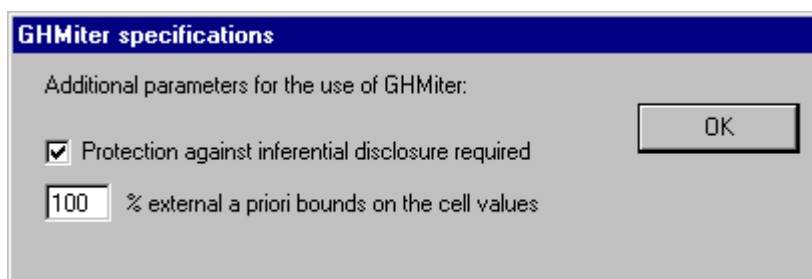
Modular

This partial method will break the hierarchical table down to several non-hierarchical tables, protect them and compose a protected table from the smaller tables. As this method uses the optimisation routines, an LP-solver is required: this will be either XPRESS or CPLEX. The routine used can be specified in the Options box, this will be discussed later.

Optimal

This method protects the hierarchical table as a single table without breaking it down into smaller tables. As this method uses the optimisation routines, an LP-solver is required: this will be either XPRESS or CPLEX. The routine used can be specified in the Options box, this will be discussed later.

It is the responsibility of the users of τ-ARGUS to apply for a licence for one of these commercial packages themselves. Information on obtaining one of these licences will be found in a 'read.me' file supplied with the software.
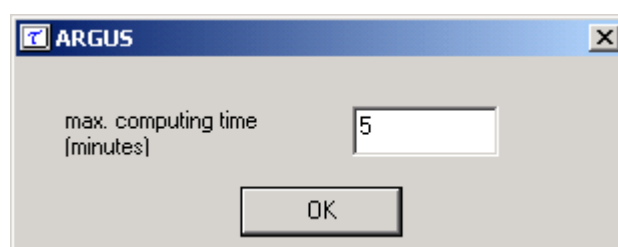
By choosing 'Suppress/Optimal' a further question is asked. The question is 'How much time do you allow the system to compute the optimal solution'.

Network

This is a Network Flow approach for large unstructured 2 dimensional tables and 2 dimensional hierarchical tables with only one hierarchy (the first variable specified). The user has the option of selecting an optimisation method (PPRN and Dykstra). Both optimisation methods are available free of an additional licence. By default the Dykstra solution is advised.

As the network solution is an heuristic to find an approximation of the real optimal solution, it cannot be expected that always an optimal solution is found. Nevertheless it is guaranteed that at least a good feasible solution is found in a relatively short time. The order in which the primaries are provided to the network algorithm could influence the solution found. Therefore three options are available to order the primaries.

Singleton Suppression

A singleton is a cell with only one contributor. Often such a cell is unsafe, due to a particular sensitivity rule. With two singletons on a row/column they can very well protect each other. However these singleton-companies can fill in their own value and so undo the suppression. The hypercube and the modular suppression have provision to prevent two singletons on a row or column.

A problem occurs if the singleton cells are not in the same row or column but it is possible for a contributor to break the suppression. In this case only the hypercube has the facility to cope. Therefore if another suppression method is required, the user must first apply the hypercube method on the singleton cells before carrying out a full suppression using the required method.

The singleton suppression is only a pre-suppression, protecting the singleton cells. Additionally one of the other suppressions has to be applied.

Choose the suppression method

After selecting one of the options, click either the Singleton or Suppress button. τ-ARGUS will run and display a protected table after informing the user of the number of cells selected for secondary suppression and the time taken to perform the operation. The secondary suppressed cells will be shown in blue.

When the user is satisfied with the table it can be saved (see section 4.5.1 for the possible formats). Press the write-table button. This is the same button as via the menu Output|Save table.

4.4.2.4 The Options at the Bottom of the table

At the bottom of this window there are a few additional options. These options will be described here.

Change View
By clicking on Change View in the Table window after clicking on Modify|ViewTable at an earlier stage the dialog box below pops up. The user can specify which variable is wanted in the row and the column. In the two-dimensional case, the table can only be transposed. In the higher dimensional case, the remaining variables will be in the layer. For these layer variables a combo-box will appear at the top of the table, where the user can select a code. This will show the corresponding slice of the table.

For a 3 dimensional table, this window is as follows:

Table summary

Pressing 'table summary' provides a table summary giving an overview of the number of cells according to their status. The example shown here refers to the case after secondary suppression has

---

been performed.

The headings in the summary window are as follows:

Freq: The number of cells in each category

 # rec: The number of observations in each category

#Holding: The number of holdings in each category (0 if holdings are not used for this table)

Sum Resp: Total cell value in each category

SumCost: The sum of the cost variable.

3 dig separator
This removes or inserts the character separating the thousands for the values in the table.

Output View
This option allows the table to be shown as it will be output, with suppressed cells (primary and secondary) replaced by a X.

4.4.3 Linked Tables

This option is available when the tables specified have at least one explanatory or spanning variable in common and the same response variable.
An example is shown.

After the tables have been computed, under the 'Modify Tables ' option, 'Linked Tables' is available. The following table appears. Clicking on 'Go' will protect the tables simultaneously, using the hypercube method.
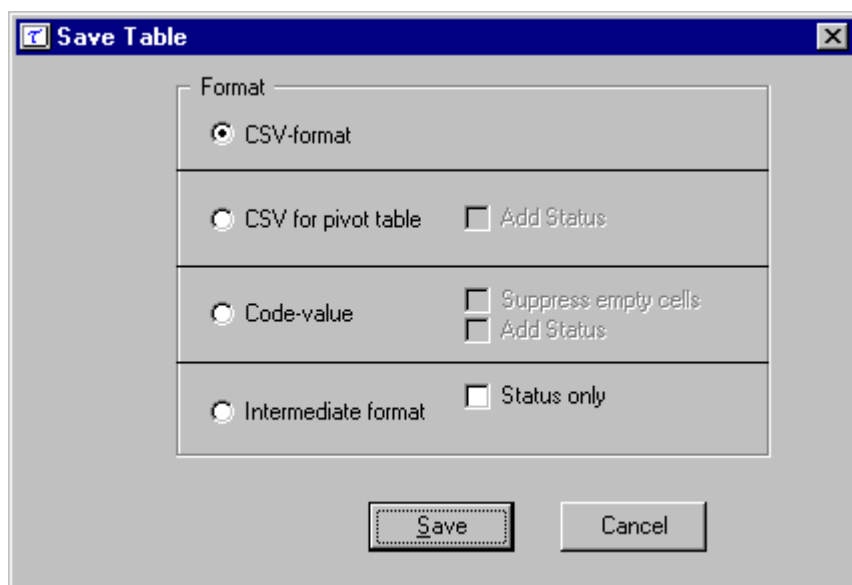

This procedure is only available when microdata is read into the program. Currently, when tabular data are entered only one table may be entered thus making the protection of linked tables impossible.

4.5 The Output menu

4.5.1 Output | Save Table

There are four options of storing the tables

1. As a CSV file. This Comma Separated file can easily be read into Excel. Please note the Excel should interpret the comma as a separator. If your local settings are different you could use the Excel option 'Data|Text to Columns', This a typical tabular output maintaining the appearance of the table in τ-ARGUS.

2. A CSV-file for a pivot table. This offers the opportunity to make use of the facilities of pivot table in Excel. The status of each cell can be added here as an option (Safe, Unsafe or Protected for example). The information for each cell is displayed on a single line unlike standard csv format.

3. A text file in the format code-value, this is separated by commas. Here, the cell status is again an option. Also empty cells can be suppressed from the output file if required. The information for each cell is displayed on a single line similar to the CSV file for a pivot table.

4. A file in intermediate format for possible input into another program. This contains protection levels and external bounds for each cell. This file could even be read back into τ-ARGUS, using the read tables option

Finally, a report will be generated to a user specified directory. This report will be shown, when the table has been written. As this is an HTML-file it can be viewed easily later.

4.5.2 Output | View Report

Views the report file which has been generated with Output|Save Table. An example of the output HTML file is shown here.
As can be seen the essential information, for somebody other than the user, about which rules have been applied to make the data safe is displayed along with details of any recoding.

τ-ARGUS Report

Table created date: 04-01-2004 time 10:36:08

Original file:	*H:\Anco\TauArgusVB\Datata\tau_testW.asc*
Meta file:	*H:\Anco\TauArgusVB\Datata\tau_testW.rda*
Table file:	*H:\Anco\TauArgusVB\Datata\test.csv*

Table generated from microdata

Table structure

Var	*Function*	*# codes*
Response var:	*Var2*	
Explanatory var 1:	*Size*	*9*
Explanatory var 2:	*Region*	*18*
Shadow variable :	*Var2*	
Cost variable :	*Var2*	

Safety Rule:
Dominance rule (Indiv. level) with n = 3 and k = 75%
Manual safety margin: 20%

GHMITER solution

GHMITER range ratio used: 0.667

Time used to protect the table: 3 sec

Summary of the table

	Status	Number of cells	Number of respondents	Response value	Cost value
1	*Safe*	*96*	*236141*	*94869102.04*	*94869102.04*
2	*Safe (manual)*	*0*	*0*	*0.00*	*0.00*
3	*Unsafe*	*12*	*52*	*12058.00*	*12058.00*
4	*Unsafe (request)*	*0*	*0*	*0.00*	*0.00*
5	*Unsafe (Freq)*	*0*	*0*	*0.00*	*0.00*
6	*Unsafe (Zero cell)*	*0*	*0*	*0.00*	*0.00*
7	*Unsafe (Singleton)*	*0*	*0*	*0.00*	*0.00*
8	*Unsafe (Singleton) (manual)*	*0*	*0*	*0.00*	*0.00*
9	*Unsafe (manual)*	*0*	*0*	*0.00*	*0.00*
10	*Protected*	*0*	*0*	*0.00*	*0.00*
11	*Secondary*	*11*	*20145*	*6204721.00*	*6204721.00*
12	*Secondary (fr. man.)*	*0*	*0*	*0.00*	*0.00*
14	*Empty*	*43*	*0*	*0.00*	*0.00*
15	*Total*	*162*	*256338*	*101085881.04*	*101085881.04*

Recoding for variable Size

Recoding tree for variable Region

CodeTree
total
.Nr
.. 1
.. 2
.. 3
.Os
.. 4
.. 5
.. 6
.. 7
.Ws
.. 8
.. 9
..10
.Zd
..11
..12
.99 (missing)

τ-ARGUS version: 3.0

4.5.3 Output | Write Batch File

The commands used in interactive mode can be saved into a file for future use. τ-ARGUS will write a batch file containing the commands necessary to achieve the current situation of the τ-ARGUS run so far. For more information on the batch facility see section 4.2.3

For example the following shows the dominance rule (n=3, k= 75) applied to the Size by Region table with Var2 as the response variable. The threshold value = 5 with a safety range = 30%. Modular secondary suppression was applied. The last line indicates that τ-ARGUS will not stop after these commands but become an interactive program.

```
<OPENMICRODATA>  "C:\Program Files\TauARGUS\data\tau_testW.asc"
<OPENMETADATA>   "C:\Program Files\TauARGUS\data\tau_testW.rda"
<SPECIFYTABLE>   "Size""Region"│"Var2"│""│""
<SAFETYRULE>     NK(3,75)│NK(0,0)│FREQ(5,30)│
<READMICRODATA>
<SUPPRESS>        MOD(1)
<GOINTERACTIVE>
```

4.6 The Help menu

4.6.1 Help | Contents

This shows the contents page of the help file and from there makes the help available.

This program has context-sensitive help.

4.6.2 Help | Options

There are a number of options, which can be changed here. Firstly, if the CPlex optimisation routine is being used, the location of the licence file can be specified here. Also, the default colours for the differently specified cells can be altered.

Within this Option box the user chooses which solver has been selected.
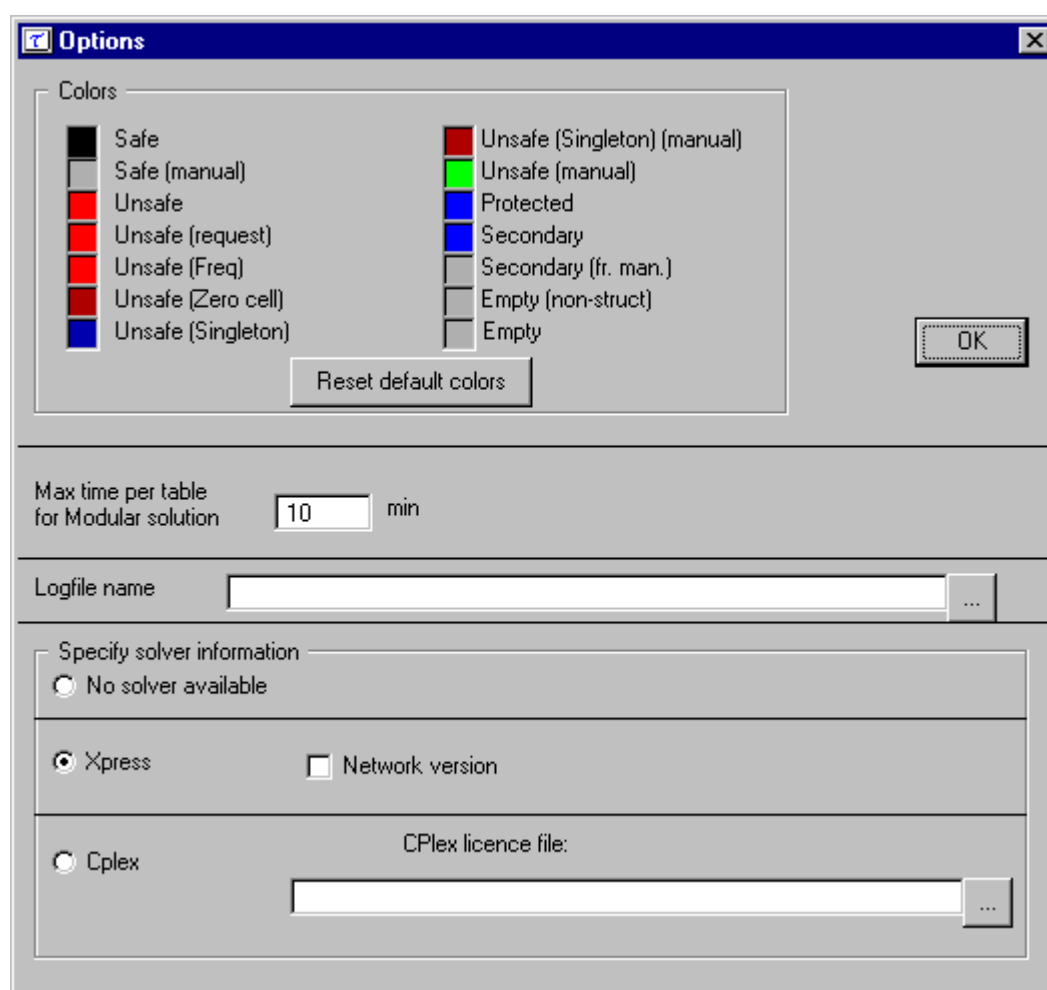
These are: No Solver, CPLEX or XPRESS. The option chosen here will determine whether or nor suppression methods based on these solvers are available. τ-ARGUS will store this information in the

registry and will use it in future runs. It is advisable but not necessary to open this window at the start of a τ-ARGUS session to ensure the correct solver has been chosen.

Also the name of the logfile (see section 4.7) can be changed here. By default it is Logbook.txt.

4.6.3 Help | About

Shows the about box.

4.7 Log file

τ-ARGUS will write a log-file. This describes among others the commands used during the runs of τ-ARGUS. If gives a log of the use of τ-ARGUS. Especially for the batch process this file could give some information about the progress of the process. Below is given a small example. Please note that new information is always added to this file. So from time to time the user should erase this file to clean his computer.
By default the logfile is the file LOGBOOK.TXT in the temp-directory. In the options window the name of the logfile can be changed for the remainder of the current session.

```
12-aug-2004 16:20:51 : Start preparing for tabulation
12-aug-2004 16:20:51 : Table 1: Size x Region | Var2
12-aug-2004 16:20:51 : Start explore file: H:\Datata\tau_testW.asc
12-aug-2004 16:20:55 : Start computing tables
12-aug-2004 16:20:56 : Compute tables completed
12-aug-2004 16:21:00 : Start Modular optimisation
12-aug-2004 16:21:09 : HITAS finished successfully
12-aug-2004 16:21:14 : Suppression has been undone
12-aug-2004 16:21:15 : Start FullJJ optimisation
12-aug-2004 16:21:17 : Full JJ optimisation finished successfully
12-aug-2004 16:21:31 : Table no: 1 has been saved
                       FileName: H:\Datata\test.csv
                       as Save table in CSVformat
```